Robert Paciorek

Komputerologia dla każdego

http://www.opcode.eu.org/

2015-10-20

Spis treści

Technika cyfrowa

Systemy POSIXowe

Kilka słów (wstępu) o programowaniu

Więcej na temat programowania

Systemy oraz instalacje kablowe inteligentnego budynku

Inne

Wstęp do techniki

Wprowadzenie

W kontaktach z technologią, byciu dobrym ”komputerowcem”, elektronikiem, … najważniejsze wydaje się być **bawienie się technologią**, eksperymentowanie, wyciąganie wszystkiego co się da z danego układu, łączenie i integrowanie różnych systemów i technologii, poznawanie nowego, głębokie poznawanie, traktowanie problemów jako wyzwań, … oraz czerpanie z tego radości. Do określenia kogoś o takiej postawie, takim podejściu do techniki, najlepszym terminem byłby **haker**, pomimo iż jest to termin różnie rozumiany, u wielu osób wywołujący negatywne skojarzenia, wydaje się on najbliższy postawie życiowej jaką starałem sie tu przedstawić (trochę więcej na stronach Wikipedii[1]).

Podobnie jest przy poznawaniu świata komputerów, elektroniki, telekomunikacji, … , przy nauce tych dziedzin bardzo istotne (jeżeli nie najistotniejsze) wydaje się otwarte podejście do stawianych problemów, **eksperymentowanie**. Dla przykładu przy poznawaniu obsługi bądź konfiguracji jakiegoś nowego narzędzia równie istotne jak zapoznanie się z dokumentacją, podręcznikiem jest wypróbowanie różnorakich opcji, ich różnych kombinacji w praktyce, poszukanie takiego sposobu konfiguracji, użytkowania (takiego stylu) jaki jest najbardziej odpowiedni dla nas i dla rozwiązywanego problemu. Nie ma większego sensu uczenia się szczegółów, konkretnych implementacji, itp na zapas. Jeżeli możemy sobie tylko na to pozwolić warto uczyć się na ”żywych” zastosowaniach (rozwiązując problemy, które gdzieś w jakiś sposób samoistnie się pojawiły); przy takiej nauce warto też pamiętać o ogromnych zasobach Sieci i na przykład www.google.com :-). W rozwiązywaniu konkretnych zagadnień najistotniejsze jest wiedzieć jak się do tego zabrać i umieć dać sobie radę.
Wydaje się że właśnie w tych dziedzinach często od wiedzy szczegółowej, ważniejsza jest szeroka **wiedza ogólna** (oraz oczywiście odpowiednie predyspozycje i chęci), wraz z umiejętnościami szukania informacji (i wiedzą gdzie należy jej szukać), sporą dozą praktyki (rzeczywistość tworzy sytuacje które nie śniłyby się teoretykowi …) i tak zwanym obyciem z techniką / technologią. ”Komputerologia” nie wymaga posiadania pełnej wiedzy na początku wykonywania zadania1, często nawet ekspert może czegoś nie wiedzieć, szukać (nieco po omacku) działającego rozwiązania i w zasadzie ma do tego prawo (tak więc różnica między uzdolnionym uczniem a dobrym ekspertem może polegać tylko na ilości doświadczenia - a ono jest w tej dziedzinie dość ważne …). Gdy napotka się jakiś problem należy próbować go rozwiązać, eksperymentować, szukać porady (ale na zasadzie ”że chcemy dostać wędkę a nie rybę”) i nie żałować na to czasu. Dla przykładu umiejętność programowania to zasadniczo jakieś podstawy i zrozumienie tego jak to działa i trochę kombinowania, oprócz tego przydatne są dobre dokumentacje do API wykorzystywanych bibliotek (w tym bibliotek standardowych) oraz podręczniki ”algorytmy i struktury danych” i ”metody numeryczne” ;-) . Poniżej spróbowałem zebrać i streścić wspominane podstawy teoretyczne (które każdy komputerowiec znać powinien …).

Podstawowym zadaniem chyba wszystkich systemów komputerowych i telekomunikacyjnych oraz ogromnej większości systemów elektronicznych jest przetwarzanie lub przekazywanie **informacji**. Obecnie jest to w znacznej większości informacja **cyfrowa** (w odróżnieniu od analogowej jest ona ”skwantowana” - sygnał będący jej źródłem zmieniający się w sposób ciągły został poddany procesowi pomiaru i zamiany na liczbę o skończonej dokładności).
Właśnie to oraz (możliwe do wprowadzenia dzięki postaci liczbowej) sumy kontrolne umożliwiają wierne kopiowanie informacji. Dzięki wprowadzeniu tego dodatkowego matematycznego poziomu w zapisie informacji odrywamy się po części od ograniczeń naszej fizycznej rzeczywistości; ”bity to nie atomy” i fizyka (w szczególności kwantowa) ich nie dotyczy.

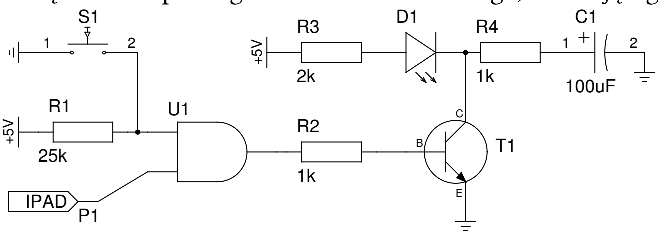
Podstawową jednostką informacji jest **bit**, mogący przyjmować dwa rozróżnialne stany nazywane zazwyczaj **zero** i **jeden**, będący odpowiedzią typu tak/nie. Przy pomocy mniejszej ilości stanów nie jest możliwe przekazanie informacji (pozornie mogłoby wydawać się że niekiedy wystarczy jeden stan - np. podnosimy n razy chorągiewkę i w ten sposób przekazujemy liczbę n, jednak tutaj też jest ukryty drugi stan - chorągiewka opuszczona). Z stanami tymi łatwo związać odpowiednie stany napięciowe na wejściach/wyjściach układów elektronicznych - np. napięcie powyżej X voltów (”jest napięcie”) oznacza 1, napięcie poniżej Y voltów (”brak napięcia”) oznacza 0.

Wydaje się że właśnie z tych dwóch powodów oraz z faktu iż **system dwójkowy** (mający tylko dwie cyfry - 0 i 1) jest systemem liczbowym o najmniejszej możliwej podstawie, system ten wraz z **logiką dwuwartościową** zyskał tak duże zastosowanie w **elektronice** cyfrowej i komputerach.

Elektronika

Elektronika (zarówno analogowa jak i cyfrowa) zajmuje się zasadniczo przetwarzaniem informacji w postaci sygnałów elektrycznych. Podstawowymi pojęciami jest tutaj prąd (przepływ ładunku) oraz napięcie (**różnica potencjałów** czyli siła ten przepływ wymuszająca). Elementy elektroniczne podzielić możemy na elementy bierne (oporniki, kondensatory, cewki, diody) - wpływające na przepływ prądu w sposób nieregulowany oraz elementy aktywne (których działanie w odróżnieniu od biernych jest regulowane na drodze sygnałów elektrycznych). W elektronice (zwłaszcza cyfrowej) często dużo ważniejszym pojęciem niż prąd jest napięcie, a dokładniej nawet tylko potencjał w odniesieniu do umownego potencjału zerowego - masy (**GND**).

Dla przetwarzania danych przez współczesną elektronikę najważniejszym z podstawowych jej elementów wydaje się być **tranzystor** - jest to element o regulowanym poprzez przyłożenie odpowiedniego napięcia oporze elektrycznym (może być zatem wykorzystywany jako elektronicznie sterowany przełącznik). Obecnie dość rzadko spotykany jest jednak jako niezależny składnik urządzenia, najczęściej występuje wraz z dziesiątkami czy nawet milionami innych tranzystorów w postaci układów scalonych. Podstawowymi dla techniki cyfrowej układami scalonymi są **bramki logiczne** realizujące poszczególne funkcje logiczne na przetwarzanych sygnałach (trochę więcej o nich, systemie dwójkowym i logice dwuwartościowej, znajduje się w artykule technika cyfrowa). Poniżej przedstawię oraz omówię schemat prostego układu elektronicznego, obrazującego podstawowe aspekty elektroniki.



[2]

U1 jest dwu wejściową bramką logiczną AND (na wyjściu zwraca stan logiczny 1 gdy na obu wejściach ma stan logiczny 1, któremu w tym wypadku odpowiada stan wysoki +5V), do jednego z wejść podłączony jest jakiś sygnał sterujący z zewnątrz (P1) do drugiego podłączony jest natomiast przełącznik S1 zwierający do masy. Rezystor R1 podłączony do +5V zapewnia iż gdy przełącznik nie jest zwarty to do wejścia bramki przyłożony jest stan wysoki (niektóre z układów scalonych mają takie rozwiązanie zaimplementowane wewnętrznie, nawet w tym wypadku nie jest on konieczny gdyż bramka ta wejścia nie podłączone traktuje jako logiczne 1), a gdy zwieramy S1 to ogranicza płynący prąd. Wyjście bramki poprzez kolejny opornik (R2), który ogranicza prąd pobierany z wyjścia układu, steruje tranzystorem NPN T1 (przewodzi on gdy napięcie miedzy bramką (B) a emiterem (E) przekroczy określoną wartość). Wprowadzenie tranzystora w stan przewodzenia powoduje przepływ prądu przez LED D1 oraz rezystor R3, który ogranicza ten prąd do wartości odpowiedniej dla D1 (oczywiście także T1 musi być dostosowany do wartości tego prądu); warto tu także zwrócić uwagę na polaryzację D1 (LED jak każda dioda przewodzi tylko w jedną stronę) oraz na to iż tranzystorem możemy sterować wiele innych układów - nie koniecznie LEDy. Kondensator C1 wraz z rezystorem R4 (a także R3) powoduje spowolnione wygaszanie D1 - kondensator rozładowuje się przez R4 gdy T1 przewodzi, natomiast po zaprzestaniu przewodzenia przez T1 kondensator ten ładuje się przez R3 i R4 podtrzymując przez pewien czas świecenie diody (dioda przygasa powoli, gdyż spada na niej napięcie w skutek procesu ładowania C1). Na koniec należy wspomnieć że przedstawiona konwencja symboli dla elementów elektronicznych nie jest jedyną.

Jednak także i bramki ustąpiły już miejsca **układom programowalnym** złożonym z tysięcy tranzystorów, bramek i tym podobnych elementów - np. programowalnym układom logicznym, procesorom czy też mikrokontrolerom (zawierającym także pamięć zmienną - dla danych programu oraz pamięć do przechowywania samego programu); sprowadzając w zasadzie elektronikę opartą na elementach dyskretnych do roli pomocniczej. Właśnie niczym mniej ani więcej tylko takim programowalnym urządzeniem elektronicznym, o zdolności do przetwarzania znacznych ilości informacji (w postaci liczb) w krótkim czasie (znacznej mocy obliczeniowej), jest współczesny **komputer**. Złożony on jest zasadniczo z procesora, pamięci operacyjnej (wraz z układem odpowiedzialnym za umożliwienie dostępu do niej procesorowi - kontrolerem pamięci) i pamięci stałej (dysk twardy itp.) służącej do przechowywania programów i danych. Niezależnie od roli jaką pełni (maszyna do pisania czy grania, serwer WWW czy też serwer nadzorujący procesy w reaktorze jądrowym, ...) każdy komputer tylko i wyłącznie wykonuje jakiś **program**.

Poprzez powyższe twierdzenie, że komputer wykonuje tylko i wyłącznie program będący ciągiem instrukcji, nie twierdzę że nie może istnieć coś takiego jak **sztuczna inteligencja**, a nawet sztuczna świadomość. Obecnie istnieją systemy zdolne do samodzielnej nauki na podstawie historycznych doświadczeń, istnieją systemy decyzyjne oraz wiele innych rozwiązań które w jakimś stopniu można by uważać za inteligentne. Z problemem sztucznej inteligencji związana jest głównie nieprecyzyjność tego terminu (istnienie wielu sprzecznych definicji - zobacz w Wikipedii[3]). Wydaje się, że nawet człowiek nie jest w stanie przekonać innej osoby o tym że jest istotą inteligentną a tym bardziej świadomą gdy tamta przyjmie inne założenie.

Oprócz opisanych fundamentów warto zainteresować się także m.in. programowalnymi matrycami elemen-

tów cyfrowych i analogowych, systemami komputerowymi do zastosowań wbudowanych, systemami sterowania, nadzoru i monitoringu, bezpieczeństwa, a także samą elektryką. O wielu z tych zagadnieniach wspominam także w podlinkowanych artykułach.

Programowanie

Program jest to ciąg instrukcji (głównie operacji matematycznych i logicznych), zapisany w języku zrozumiałym dla danego procesora (kod maszynowy powstający w wyniku kompilacji kodu stworzonego w języku wyższego poziomu). Wśród tych instrukcji szczególne miejsce zajmują instrukcje skoków (warunkowych i bezwarunkowych), one właśnie umożliwiają warunkową realizację fragmentów programu oraz warunkowe powtarzanie (**iterację**) fragmentów programu. Za szczególny przypadek instrukcji skoku można uważać instrukcję wywołania procedury, powodującą przekazanie stosownych informacji i rozpoczęcie wykonywania podprogramu (poprzez warunkowe wywoływanie tej samej procedury z jej wnętrza możemy otrzymać "konkurencyjny" do iteracji sposób powtarzania fragmentu programu - **rekurencję**.

Podstawą działania każdego programu jest jakiś **algorytm**, jest to pojęcie trochę bardziej abstrakcyjne niż program (zwłaszcza rozumiany jako kod maszynowy) - ogólnie mówiąc jest to skończony ciąg czynności prowadzących do zrealizowania jakiegoś zadania. Jak już wspomniałem oprócz kodu maszynowego (złożonego z numerów rozkazów i ich parametrów) są również języki programowania. Najbardziej podstawowym jest **asembler**, który pozwala używać łatwiej zrozumiałych dla człowieka nazw zamiast numerów instrukcji, rejestrów (w których procesor przechowuje wartości którymi operuje), ... Jednak jego instrukcje odpowiadają niemalże w 100% instrukcjom procesora (kodowi maszynowemu), dlatego też jest bardzo silnie zależny od danej architektury sprzętowej (modelu procesora), powoduje to problemy z przenoszeniem programów pisanych w asemblerze na inne architektury, to było jednym z powodów stworzenia **języków wyższego poziomu**. W językach takich treść algorytmu wyraża się przy pomocy poleceń danego języka będących bytami bardziej abstrakcyjnymi od instrukcji procesora (np. pojęcie zmiennej i operowanie na niej a nie na miejscach w pamięci i rejestrach). Obecnie asemblera używa się w zastosowaniach wymagających dużej wydajności (zarówno w oddzielnych aplikacjach jak i fragmentach większych programów) bądź dla urządzeń o niewielkich rozmiarach pamięci.

Najistotniejszym obecnie językiem programowania wydaje się być **język C** (a także jego "poprawiona", obiektowa wersja C++). Program w C składa się zasadniczo z funkcji (szczególną rolę pełni funkcja `main()` od której zaczyna się wykonywanie programu), oprócz tego mamy zmienne (także tablice i struktury), instrukcje sterujące (pętle, warunki - for, while, if, switch, ...), a także instrukcje preprocesora (umożliwiające dostosowanie kodu naszego programu przed kompilacją ...). Język ten umożliwia nam także operacje na adresach w pamięci pod którymi przechowujemy dane (wskaźniki), pozwala na operacje matematyczne, logiczne, binarne, posiada dość rozbudowaną bibliotekę standardowych funkcji, ... Z kolei podobny do niego C++ umożliwia ponadto programowanie obiektowe (class, ...) oraz ma dodatkowo własną bibliotekę standardową. Poniżej zamieszczam krótki program obrazujący koncepcję programowania w C:

```
// funkcja główna od której zaczyna się wykonywanie programu, może przyjmować argumenty
// odpowiadające za wartości przekazane w linii poleceń oraz zwracać kod powrotu
main() {
  // zmienna typu char:
  char a=0;

// zmienna tablicowa (czyli N zmiennych danego typu zapisanych w kolejnych komórkach
 pamięci):
  char tab[3] = {54, 0x2A, 124}; // warto zwrócić uwagę na zapis szesnastkowy

// wskaznik - czyli zmienna przechowująca adres innej zmiennej / funkcji / struktury danych
  // w tym wypadku dwa wskazniki do zmiennych typu char
  char *wskA, *wskB;

// nadanie im wartości będącej adresem a i adresem początku tablicy tab
  wskA=&a; wskB=tab; // warto zwrócić uwagę na operator wyłuskania adresu zmiennej

// powtorzenie 10 razy ... ale nie przez pętlę for:
  while (a++ > 10) { // zwiększamy a o 1 po sprawdzeniu warunku ...
    // wypisanie:
    printf("obieg petli numer: %i\n", a);
```

```
    // gdy a jest 3:
    if (a==3) {
     printf("\ta = %i \n\ttab[0] = %i \n\ttab[1] = %i\n", *wskA, tab[0], *(wskB+1));
       // warto zwrócić uwagę na operację wyłuskania wartości z pod wskaźnika
       // oraz artmetykę wskaźnikową
    }
  }
  /* na koniec należy wspomnieć iż wskaźniki są dość niskopoziomowym mechanizmem
       umożliwiającym ”rozszerzenie” zakresów zmiennych poza najbliższe klamerki -
       np. funkcja może przyjmować wskaźnik i modyfikować bezpośrednio zmienną
       z kodu który ją wywołał ... normalnie nie byłaby ona dla niej dostępna */
}
```

Warto jeszcze wspomnieć o istnieniu całej masy innych języków programowania oraz o tym, że podstawową różnicą między nimi jest stosowany formalizm i nazwy funkcji bibliotecznych - na ogół większych różnic w samej koncepcji programowania nie ma (są jednak pewne ”wyjątkowe” języki, np. erlang). Szczególnie zachęcam do zapoznania się z wybranymi programami prezentującymi podstawy programowania w C/C++ w działaniu: podstawy.c, petle_warunki.c, in_out.c, numeracja_ip.c, wskaznikologia.c. Polecam również artykuły poświęcone **skryptowym językom programowania**, stanowiącym niezbędnik administratora systemów Unix'owatych - awk (tu warto wspomnieć że np. w odróżnieniu od C nie ma konieczności deklarowania w nim typów zmiennych, a tablice indeksowane są dowolnym napisem ...) i (ba)sh (język ten w związku ze swoją specyfiką wyróżnia się tym iż większość komend (poza podstawowymi instrukcjami sterującymi - takimi jak if, while, ...) jest realizowana przez osobne programy - komendy). Warto także zapoznać się z artykułami poświęconymi bazom danych, php oraz innym językom programowania. Natomiast więcej na temat algorytmów znaleźć można w Wikipedii[4].

Oprócz opisanych fundamentów warto zainteresować się także m.in. programowaniem obiektowym, systemami bazodanowymi, sieciami neuronowymi, programowaniem współbieżnym. O wielu z tych zagadnieniach wspominam w podlinkowanych wyżej artykułach.

Telekomunikacja

Oprócz przetwarzania informacji liczne systemy zajmują się jej przekazywaniem. Aby mógł zajść proces komunikacji musi istnieć jakieś medium transmisyjne, które będzie odpowiedzialne za fizyczne przekazanie sygnału zawierającego informację. W **telekomunikacji** przekaz ten odbywa się zazwyczaj z wykorzystaniem zjawiska przepływu prądu przez przewodnik (tradycyjna łączność kablowa), bądź rozchodzenia się fali elektromagnetycznej (łączność radiowa, światłowody, ...). Bardzo istotne jest dopasowanie parametrów medium transmisyjnego, urządzeń nadawczych i odbiorczych do sygnału który zamierzamy transmitować, na przykład w przypadku kabli jest to częstotliwość którą są one w stanie przenosić bez zakłóceń (głównie ona odpowiada za szybkość transmisji i najczęściej jest rzędu kilkuset MHz) oraz impedancja (wyrażana w omach, sygnały które transmitujemy są sygnałami zmiennymi dlatego jest ona ważnym parametrem przewodu ...). Jednak pomimo to zawsze należy mieć świadomość, że to tylko kabel, który przewodzi elektrony i niezależnie od parametrów jakoś będzie to robił ... .

W tradycyjnych systemach telekomunikacyjnych sygnał był bezpośrednio informacją, a jego dotarcie do odbiornika oznaczało że informacja jest przeznaczona dla niego (telefonia) bądź dla każdego (radio, telewizja - systemy rozgłoszeniowe). Obecnie coraz większą rolę odgrywają komutowane **sieci pakietowe**, gdzie informacja nie jest tak ściśle związana z sygnałem. Dla przykładu w telefonii zrealizowanie każdego połączenia wymagało zestawienia go - odpowiedniego przełączenia styków w centralach (komutacji łączy); w Internecie informacja podążać może różnymi drogami (nie jest tworzona droga specjalnie dla danego połączenia) i na podstawie adresu odnaleźć host docelowy (mamy tu doczynienia z komutacją pakietów). W sieciach pakietowych właściwa informacja (jeżeli jest zbyt długa) dzielona jest na części, do każdej części dodawane są informacje kontrolne - nagłówek (zawierający m.in. adres odbiorcy). Tak utworzone pakiety podróżują niezależnie od siebie przez sieć do odbiorcy (bez zestawiania w tym celu specjalnego fizycznego połączenia).

Bardzo istotnym elementem jest adresacja (jest ona niezbędna aby informacja mogła być kierowana do konkretnego hosta - inaczej mamy sieć rozgłoszeniowa (np. telewizja)): każdy host w sieci musi posiadać unikalny (w ramach tej sieci) adres (chyba zawsze o ustalonej długości - nie jestem w stanie zagwarantować ze ktoś, gdzieś nie stworzył protokołu pakietowego z adresami zmiennej długości), często wydziela się również **podsieci**, posiadają one swój adres (na ogół jest to wspólna część adresu wszystkich hostów podsieci dopełniona zerami), adres określający ile adresów jest w podsieci () oraz adres na który reagować powinien każdy host w tej podsieci (adres rozgłoszeniowy). W przypadku chyba najpopularniejszych obecnie sieci **IP** wielość podcieci była dawniej determinowana przez klasy

numerów IP, natomiast współcześnie określana jest przez **maskę podsieci**, która po wykonaniu binarnego AND z dowolnym adresem hosta tej sieci da adres sieci. W związku z tym iż binarnie maska składa się z nieprzerwanego ciągu jedynek oraz nieprzerwanego ciągu zer, jest ona często zapisywana jest w postaci prefixowej, czyli liczby bitów mających wartość jeden. W sieciach IP (oprócz samego Internet Protocol - odpowiedzialnego za adresację itp) należy także wspomnieć o protokołach takich jak: ICMP - zapewniającym przesył komunikatów technicznych, oraz **UDP** czy TCP - umożliwiających przesył danych poprzez sieć IP (odpowiedzialne one są m.in. za adresację usług na danym hoście poprzez numery portów, **TCP** czuwa także nad poprawnością transferu - kontroluje czy dane dotarły i czy dotarły nieuszkodzone). Protokół IP oprócz najbardziej dla niego typowej transmisji jeden do jednego (unicast) umożliwia transmisję rozgłoszeniową (broadcast - jeden do wszystkich w sieci) i multicast (jeden do wielu).

Oczywiście sieci teleinformatyczne to nie tylko IP - należy koniecznie wspomnieć o protokołach warstw wyższych - takich jak HTTP (sieć Web), SMTP (e-mail), … - zapewniających realizację typowych usług sieciowych oraz protokołach związanych z konkretną realizacją sprzętową sieci - np. ethernet, czyli najpopularniejszy standard sieci lokalnych (LAN). Ze sprzętową realizacją sieci wiąże się m.in. pojęcie takie jak **topologia sieci** - czyli zasada łączenia hostów (**gwiazda** - wszystkie do jednego punktu, drzewo - wiele gwiazd łączonych w gwiazdy, **magistrala** - od jednej linii każdy host, …). Najpopularniejszą obecnie topologią (przekładającą się także na inne instalacje - np. elektryczną, gdzie jednak dalej istotna jest magistrala) jest topologia gwiazdy, gdzie mamy doczynienia z punktami dystrybucyjnymi (w profesjonalnym podejściu realizowanymi w oparciu o szafy krosownicze i specjalnie przeznaczone do tego celu pomieszczenia) od których odchodzą trasy kablowe. Przy realizacji systemów telekomunikacyjnych (i innych teleinformatycznych oraz elektrycznych) warto pamiętać o stosownej dokumentacji tak przebiegu tras kablowych, jak i samej instalacji oraz punktów dystrybucyjnych a także przejrzystości i opisaniu samych instalacji. Zaznaczyć też należy różnice między systemami dedykowanymi (np. routery sprzętowe) a systemami ogólnego przeznaczenia używanymi w takich zastosowaniach - te drugie są na ogół dużo bardziej elastyczne, natomiast dedykowane są bardziej wydajne i często droższe … osobiście bardziej cenię elastyczność. Warto tutaj także zwrócić uwagę na "nie informatyczne" bezpieczeństwo takich instalacji obejmujące takie zagadnienia jak zabezpieczenia przeciw pożarowe, przeciw przepięciowe oraz monitoring i kontrolę dostępu do pomieszczeń przeznaczonych na wspomniane instalacje. Więcej o fizycznych aspektach budowy sieci można w artykule systemy oraz instalacje kablowe.

Oprogramowanie

Oczywiście aby się zajmować, bawić komputerami, elektroniką, sieciami telekomunikacyjnymi konieczna jest wiedza na temat samej obsługi komputerów - zwłaszcza systemów unix'owatych. W tych systemach poszczególne osoby pracują na własnych kontach (identyfikowanych loginem i chronionych hasłem), szczególnym przypadkiem jest konto super użytkownika "**root**" - służy ono do czynności administracyjnych i umożliwia zrobienie z systemem (niemalże) wszystkiego. Standardowo stosowany jest też znany z adresów e-mail zapis z małpą - umożliwiający globalną identyfikację użytkownika - user@host.domena. Z użytkownikami związane są także uprawnienia - definiowane (w podstawowym wariancie) niezależnie dla właściciela, grupy i wszystkich innych w dziedzinach odczytu, zapisu i wykonania.

Powłoka tekstowa oparta jest na poleceniach (będących najczęściej osobnymi programami) do których przekazywane są opcje i argumenty (odzielane są one - zarówno od siebie jak i od nazwy programu spacjami), opcje na ogół zaczynają się od myślnika (jednoliterowe) bądź dwóch myślników (wieloliterowe), ale są od tego wyjątki. Informacje o prawie każdym programie, większości plików konfiguracyjnych oraz funkcjach biblioteki standardowej C można znaleźć w podręczniku systemowym (polecenie **man**). Za najważniejsze komendy unixa należałoby uznać: cd, mkdir, ls, ln, cp, mv, rm, vi (tu parę najistotniejszych komendy tego edytora: [Esc] :w :q :wq :q! i dd yy p), tar, gzip, grep, less, more, wc, cut, find, grep, du, man, info; trzeba także wspomnieć o przekierowywaniu standardowego wyjścia programu do innego programu (pomiędzy poleceniami |), do pliku (> nazwa_pliku - w trybie nadpisania lub » nazwa_pliku w trybie dopisywania) oraz pobieraniu standardowego wejścia z pliku (< nazwa_pliku). Powłoka umożliwia także pracę w trybie wsadowym (nie interaktywnym) - jest to już wspomniany język skryptowy bash/sh.

Systemy unixowe charakteryzują się także w miarę jednolitą **strukturą katalogów**, zaczynających się od katalogu głównego / w którym znaleźć można m.in.: bin - programy, etc - pliki konfiguracyjne, lib - biblioteki, tmp - pliki tymczasowe, home - katalogi domowe, var/log - **logi** systemowe (czyli informacje o tym co się działo), … . Jak wspomniałem **pliki konfiguracyjne** umieszczone są w etc (lub katalogach domowych użytkowników), są to prawie zawsze pliki tekstowe, jednak (prawie) każdy program ma swój format, jedyną w miarę uniwersalną rzeczą jest to iż linia zaczynająca się od # (na ogół) jest traktowana jako komentarz. Więcej o unixach (głównie Linuxie, a konkretniej Debianie) w artykułach: podstawy POSIX (w tym podstawowe komendy), porady konfiguracyjne, oprogramowanie użytkowe oraz multimedia i (La)TeX; a także w już wspomnianych artykułach dotyczących programowania i zagadnień sieciowych.

Podsumowanie

Jak już po części wspomniałem (przy językach wyższego poziomu) współczesna technika umożliwia korzystanie z komputera, a nawet tworzenie oprogramowania bez znajomości **niskopoziomowych mechanizmów** działania tych urządzeń. Wydaje się jednak, że aby zrozumieć współczesną technikę, być dobrym programistą, administratorem, … poznanie tych mechanizmów wydaje się niezbędne i jest też niewątpliwie bardzo przydatne gdy chcemy zrobić coś "bliżej sprzętu" … .
Przy robieniu czegokolwiek warto pamiętać o standardach i normach technicznych, one naprawdę są (w zdecydowanej większości) tworzone po to aby ułatwić wszystkim życie (jakby wyglądał Internet gdyby każdy portal używał własnych standardów e-mail i www …). Należy też tak tworzyć, aby to co zrobiliśmy, napisaliśmy było przejrzyste i zrozumiałe dla innych oraz (za parę miesięcy czy lat) dla nas samych - aby kod był czytelny, dobrze skomentowany, a cały projekt posiadał dobrą i szczegółową **dokumentację** (wiem że często się nie chce, ale naprawdę warto …). Należy się także wystrzegać rozwiązań przekombinowanych, stanowiących przerost formy nad treścią (trzymać się reguły KISS[5]). Warto także zwrócić uwagę, że zanim zaczniemy coś pisać warto poszukać czy nie ma czegoś o podobnej funkcji, bądź nie ma jakiś **bibliotek**, które zrobią za nas połowę pracy - nie zawsze warto odkrywać Amerykę na nowo ;-) .

Przy administracji różnorakimi systemami ważną kwestią jest ich **bezpieczeństwo**, jednak należy pamiętać, że system ten ma też do czegoś służyć poza tworzeniem fortecy dla samej sztuki … . Uważam iż ważniejsze od tworzenia kolejnych super zabezpieczeń jest zastanowienie się nad tym jak bardzo strategiczne znaczenie ma ten system, jakie dane ma przechowywać, przetwarzać i dostosowanie zabezpieczeń do roli ochranianego systemu (podobne podejście stosuje się powszechnie przy tworzeniu systemów alarmowych). Niekiedy być może lepszym rozwiązaniem jest opracowanie procedury szybkiego odtworzenia systemu i danych niż stworzenie systemu zabezpieczeń, które uniemożliwią wygodne korzystanie z danego systemu (podobnie jak na ogół nie wydaje się sensowne szyfrowanie wiadomości e-mail typu: "wrócę na obiad o 16:30" …).

W ten sposób doszliśmy do końca tego opowiadania będącego wstępem teoretycznym do zagadnień elektroniki, komputerów, telekomunikacji, … (przedstawiającego kilka ważnych pojęć i terminów z zakresu podstaw tych dziedzin oraz trochę bardziej filozoficzne podejście do ich zrozumienia). Wszystkich czytelników zachęcam do zapoznania się z dolinkowanymi materiałami, innymi materiałami w moim serwisie[6] (głównie o charakterze praktycznym, z ciekawostkami i niezbędnymi informacjami teoretycznymi) i w Sieci (0.0.0.0/0 ;-)) oraz przede wszystkim do dużej porcji obycia z technologią, **praktyki** - bo to właśnie ona w tym wszystkim jest najistotniejsza (a to niestety wymaga dużo czasu …). Dobrym pomysłem wydaje się znalezienie pomysłu na ciekawy, interdyscyplinarny (programowanie, elektronika, unix, tcp/ip, …) projekt i rozpoczęcie jego realizacji (na którą składałoby się na przykład zbudowanie niewielkiej sieci wraz z konfiguracją serwera, zbudowanie modułu elektronicznego z mikrokontrolerem, oprogramowanie tego wszystkiego - w C i nie tylko …). To też niestety nie jest proste - na ogół trudno znaleźć coś ciekawego, nowego do zrobienia (gdyż większość programów napisano, układów zrobiono, itd.), dlatego też wspomniane bawienie się technologią, zajmowanie się przedstawioną mieszanką informatyki, elektroniki i telekomunikacji (możnaby to określić terminem **"komputerologia[7]"**, gdyż współcześnie dziedziny te są bardzo silnie oparte na technologiach komputerowych, a lepszego terminu chyba nie ma …) traktować można jako pewien styl, ale trudno jako zwykłe hobby.

Ważne jest aby robić to co się lubi - zawsze lepiej być dobrym sieciowcem z powołania niż kiepskim koderem (i na odwrót). Dokładny kierunek kształcenia nie jest bardzo istotny - lepiej żeby nie był zbyt odległy, ale i tak masy rzeczy trzeba się uczyć samemu. Warto mieć szersze spojrzenie na branże - to że specjalizujesz się np. w sieciach nie oznacza iż podstawy programowania (zarówno web-owego jak i systemowego) czy też elektroniki i innych pokrewnych zagadnień są zbędne - wręcz przeciwnie mogą być dodatkowym atutem (nie tylko w CV ale i w samej pracy). To czego się nauczysz jest Twoje, ale zdobytą wiedzę, doświadczenie warto notować (gdyż zdarza się ponownie odkrywać Amerykę i ze zdziwieniem stwierdzać "to ja się tym zajmowałem 10 lat temu???").

Słowa kluczowe:

bawienie się technologią - haker - eksperymentowanie - wiedza ogólna - informacja - cyfrowa - bit - zero - jeden - system dwójkowy - logika dwuwartościowa - elektronika - różnica potencjałów - GND - tranzystor - bramki logiczne - układy programowalne - komputer - program - sztuczna inteligencja - iteracja - rekurencja - algorytm - asembler - języki wyższego poziomu - język C - skryptowe języki programowania - telekomunikacja - sieci pakietowe - podsieci - maska podsieci - IP - UDP - TCP - topologia sieci - gwiazda - magistrala - root - man - strukturą katalogów - logi - pliki konfiguracyjne - niskopoziomowe mechanizmy - dokumentacja - biblioteki - bezpieczeństwo - praktyka - "komputerologia".

Dla zainteresowanych podam iż zbiór ten został uzyskany w wyniku polecenia: `awk 'BEGIN {RS="<strong>"; FS="</"} NR!=1 {printf("%s - ", $1)} END {print ""}'` i ręcznej edycji form gramatycznych (niestety język polski jest mało przyjazny automatycznemu przetwarzaniu).

Podstawy elektroniki

Elektronika[8] zajmuje się wytwarzaniem i przetwarzaniem sygnałów w postaci prądów i napięć elektrycznych. Zjawisko prądu[9] związane jest z przepływem ładunku (z uporządkowanym ruchem nośników ładunku), aby wystąpiło konieczna jest różnica potencjałów[10] (napięcie U) pomiędzy końcami przewodnika, prowadzi ono do neutralizacji tej różnicy. Dlatego dla podtrzymania stałej różnicy potencjałów konieczne jest istnienie źródeł prądu, prowadzących do rozdzielania ładunków dodatnich od ujemnych. Wyróżnić można dwa idealne rodzaje źródeł energii elektrycznej[11]:

- źródło napięciowe - zawsze daje taką samą różnicę potencjałów,
- źródło prądowe - zawsze daje takie samo natężenie prądu[12]
  (natężenie prądu I, zwane też prądem definiowane jest jako ilość ładunku przepływająca w jednostce czasu).

Podstawowymi prawami, z których korzysta się w elektronice są prawa Kirchhoffa:

- pierwsze - suma prądów wpływających i wypływających (przy czym te bieżemy z innym znakiem) do dowolnego węzła sieci jest zero,
- drugie - suma spadków i wzrostów napięć na wszystkich elementach obwodu zamkniętego (oczka obwodu) jest zero.

Oprócz nich mamy też prawo Ohma, które jest stosowane dla elementów o charakterystyce liniowej - spadek napięcia to prąd płynący przez dany element razy charakterystyczny dla niego stały opór[13] (stałość oporu nie jest prawdą dla wielu elementów półprzewodnikowych, a nawet żarówek). Mamy też prawa szeregowego (opór całości jest suma oporów) i równoległego (opór całości jest odwrotnością sumy odwrotności oporów) łączenia oporów.

Napięcie, potencjał i masa (GND)

Mówiąc o napięciach (zwłaszcza stałych) często mamy tak naprawdę na myśli potencjały (np. oznaczenia na schematach +5V). Zawsze należy pamiętać względem czego !!! (w związku z swobodą doboru stałych całkowania do wszystkich potencjałów możemy dodać dowolna wartość i fizyka się nie zmienia, różnica potencjałów jest na to odporna). Dlatego też w elektronice dość często stosuje się pojęcie wspólnej masy[14] - GND (często nawet nie rysowanej na schematach), będącej potencjałem odniesienia.

W sporej liczbie układów elektrycznych potencjał GND równy jest potencjałowi ziemi (np. instalacja elektryczna TN, gdzie w jednej z dziurek mamy stały potencjał 0 (ziemia), a w drugiej potencjał zmienny). Zagadnienie wspólnej masy ma też kluczowe znaczenie przy łączeniu urządzeń elektronicznych - kablami sygnałowymi nie powinien płynąć prąd związany z różnica potencjałów mas (jednak często ona występuje i dlatego np. w wtyku LPT koło 1/4 pinów to masa).

W tym miejscu należy także wspomnieć o (idealnych) źródłach prądowych i napięciowych: Na idealnym źródle prądowym może odłożyć się dowolne napięcie i zawsze płynie przez nie prąd ustalony przez wartość tego źródła. Przez idealne źródło napięcia można przepuścić dowolny prąd i zawsze napięcie pomiędzy jego zaciskami pozostaje stałe i określone wartością źródła.

Impedancja

Impedancja[15] jest to "uogólnienie" pojęcia oporu dla obwodów prądu zmiennego. Obowiązują dla niej takie same zasady przy łączeniu elementów obwodu - gdy szeregowo to dodajemy impedancje gdy równolegle to odwrotność impedancji wypadkowej jest sumą odwrotności impedancji składowych. Przy skorzystaniu z tej wartości prawdziwe jest dla nich prawo Ohma: $u(t) = i(t)Z$. Część rzeczywistą impedancji nazywamy rezystancją, a część zespoloną reaktancją. Stosunek reaktancji do rezystancji jest równy tangensowi kąta przesunięcia fazowego.

opór: $Z = R$

pojemność: $Z = \frac{1}{j\omega C}$ (dla prądu stałego jest to przerwa w obwodzie: $Z = \infty$)

indukcyjność: $Z = j\omega L$ (dla pradu stałego jest to zwarcie: $Z = 0$)

gdzie: $j = \sqrt{-1}$, a $\omega = 2\pi f$ to częstotliwość kołowa (pulsacja)

Należy pamiętać iż rzeczywiste elementy posiadają wielkości pasożytnicze (a więc są impedancjami złożonymi), które w pewnych okolicznościach mogą mieć znaczenie.

Równoważność gwiazda-trójkąt

Każdy obwód typu trójkąt (trzy punkty - A, B, C połączone rezystorami AB, BC, CA) może zostać zamieniony na układ gwiazdy (czery punkty - x, a, b, c połączone rezystorami xa, xb, xc) i na odwrót. Aby obliczyć rezystancje odpowiadającejn konfiguracji zapisujemy uklad równań wyrażających równość rezystancji pomiędzy AB i ab, AC i ac, BC i bc. Równania te mają postać: $R_{xa} + R_{xb} = (R_{ab} * (R_{bc} + R_{ca}))/(R_{ab} + R_{bc} + R_{ca})$. Należy pamiętać że w trójkącie jest to równoległe połączenie R_{ab} z sumą $R_{bc} + R_{ca}$.

Dzielnik napięcia i prądu

Dzielnik napięcia uzyskujemy łącząc szeregowo źródło prądowe wraz z dwoma opornikami. Poprzez dobór wartości oporników możemy uzyskać na wyjściu (węźle łączącym ze sobą oporniki) dowolną wartość napięcia (oczywiście mniejszą od napięcia wejściowego). Napięcie na wyjściu takiego dzielnika (rezystorze R_2) wynosi $U_{wy} = U_{we}\frac{R_2}{R_1+R_2}$. Rezystancja zastępcza takiego dzielnika wynosi $R_z = R_1||R_2 = \frac{R_1 R_2}{R_1+R_2}$. Mianem sztywnego dzielnika określa się taki dzielnik który charakteryzuje się odpowiednio małymi zmianami napięcia przy zmianach prądu co jest równoważne temu aby prąd płynący w dzielniku był 5 do 10 razy większy niż prąd z niego odbierany.

Wygodną metodą doboru wartości rezystancji celem uzyskania sztywnego dzielnika jest przyjęcie porządnej tolerancji napięcia $U_{tol_{max}}$(dla założonego maksymalnego prądu odbieranego $I_{odb_{max}}$). Następnie zapisujemy warunek $R_z \leq \frac{U_{tol_{max}}}{I_{odb_{max}}}$, a jako że $R_z < \min(R_1, R_2)$ to poprzedni warunek określa nam wartość mniejszego z oporów w dzielniku. Wartość drugiego oporu możemy uzyskać z zależności $R_1 = \frac{U_{tol_{max}}}{I_{odb_{max}}} \cdot \frac{U_{we}-U_{wy}}{U_{wy}}$ (gdy $U_{wy} < U_{we}/2$) lub $R_2 = \frac{U_{tol_{max}}}{I_{odb_{max}}} \cdot \frac{U_w y}{U_{we}-U_{wy}}$ (w przeciwnym wypadku).

Dzielnik prądu uzyskujemy poprzesz równoległe połączenie źródła prądowego z dwoma opornikami.

Obwody prądu zmiennego

Celem łatwiejszego obliczania obwodów prądu zmiennego możemy rozdzielać je na część dla prądu stałego (DC) i zmiennego (AC). Dla prądu stałego: źródło napięcia zmiennego i cewka jest zwarciem, kondensator jest przerwą w obwodzie. Dla prądu zmiennego: źródło napięcia stałego jest zwarciem, źródło prądu stałego jest rozwarciem, kondensator (jeżeli ma bardzo dużą / "nieskończoną" pojemność) jest zwarciem. Należy szczególną uwagę zwrócić na powstające po zastosowaniu takich trików łączenia elementów - często łączenia wyglądające na szeregowe są łączeniami równoległymi. Należy także pamiętać o powstającej "drugiej gałęzi" prowadzącej również do masy.

Stałą czasową $\tau = RC$ wyznacza pojemność kondensatora i jego oporowe otoczenie (rezystory które "widzi" kondensator i to w jaki sposób je "widzi" - w szczególności rezystory położone po obu stronach kondensatora mogą być połączone szeregowo poprzez masę). $1/\tau$ odpowiada dolnej lub górnej (zależnie od układu) częstotliwości granicznej filtra górno lub dolno przepustowego. Należy mieć świadomość że dla prądu stałego każdy (sprawny) kondensator zachowuje się jak rozwarcie oraz że dla odpowiednio dużych częstości (zależnych od pojemności kondensatora - im większa tym niższe te częstotliwości) kondensator przewodzi prąd zmienny praktycznie bez strat.

Zobacz w Sieci: Teoria obwodów[16] Teoria obwodów (en)[17], Metody analizy obwodów liniowych (Teoria obwodów) - dr inż. Wojciech Meller[18].

Metoda superpozycji

Pozwala ona na rozdzielenie obwodu zawierającego kilka źródeł prądu i napięcia. Obwod rozkładamy na tyle przypadków ile mamy źródeł niezależnych, w każdym z nich pozostawiamy tylko jedno (inne) źródło niezależne. Pozostałe źródła zastępujemy zwarciem (źródła napięciowe) albo rozwarciem - źródła prądowe (z punktu widzenia źródła prądowego źródło napięcia jest zwarciem - punkt potencjału V jest równoprawny punktowi potencjału masy). W każdym obwodzie znajdujemy składową (pochodzącą od niepominiętego źródła) szukanego prądu lub napięcia. Należy też zwrócić uwagę na kierunki napięć (ich znak) - prąd wypływający z rezystancji oznacza ujemne napięcie na tejże. Całkowity szukany prąd lub napięcie jest sumą składowych.

Zasada Thevenena i Zasada Nortona

Pierwsza z nich mówi iż dowolny układ źródeł i oporów liniowych może być zastąpiony pojedyńczym idealnym źródełm napięciowym (o napięciu równym napięciu na zastępowanym układzie gdy jest rozwarty - nie wpływa, ani nie wypływa z niego prąd) połączonym szeregowo z rezystancją zastepczą (obliczną po zastąpieniu wszystkich źródeł napięciowych zwarciami, a prądowych rozwarciami). Druga mówi o możliwości zastapienia takiego układu idealnym źródłem prądowym (o wartości prądu obliczanej przy zwartych zasiskach zastępowanego układu) z dołączoną równolegle rezystancją (obliczaną tak jak w poprzednim przypadku).

Rezystancja wejściowa/wyjściowa układu

Jest to wartość niezależna od prądu wejściowego (prądu pobieranego zawsze przez dany układ) i zdefiniować możemy ją jako iloczyn przyrostu napięcia do przyrostu prądu.

Elementy bierne

Opornik[19] - wprowadza do obwodu dodatkowy opór R (równy swojej rezystancji), powodując powstanie napięcia na jego końcach równego I*R. Obok rezystancji istotnym parametrem jest maxymalna moc jaką możemy wydzielić na pojedyńczym elemencie.

Kondensator[20] - gromadzi ładunek elektryczny (magazynuje energię w polu elektrycznym) i może go oddawać do obwodu. Można powiedzieć że kondensator magazynuje napięcie, czas potrzebny do zmiany napięcia na kondesatorze dany jest zależnością: $\Delta T = \frac{C \cdot \Delta U}{I}$. Podstawowym parametrem jest pojemność i maksymalne napięcie, które możemy na nim odłożyć. Oprócz tego w pewnych wypadkach należy zwrócić uwagę na parametry takie jak maksymalny prąd przewodzenia (kondensator przewodzi prądy zmienne!), impedację (ESR), liniowość zastosowanego dielektryka, itp.

Cewka[21] - magazynuje energię w polu magnetycznym (gromadzi prąd). Po odłączeniu zasilania chce ona wyrzucić z siebie prąd (a nie tak jak kondensator ma po prostu na sobie jakiś ładunek), dlatego należy uważać na przebicia w układach kluczujących cewkę. Czas potrzebny zmianę zmagazynowanego w cewce prądu dany jest zależnością: $\Delta T = \frac{L \cdot \Delta I}{U}$. Wyróżnia się przede wszystkim cewki: powietrzne, na rdzeniu żelaznym i na rdzeniu ferrytowym. Ferryt bardzo silnie zwiększa indukcyjność cewki, ale ogranicza od góry częstotliwość pracy cewki. Ponadto przy odpowiednio dużym prądzie może dochodzić do nasycenia rdzenia, które objawia się tym że zaczyna in hamować wzrost pola zamiast go poprawiać.

Przełącznik/styk - element służący do załączania lub odłączania prądu. Powszechnie spotyka się styki chwilowe (monostabilne) zwierne (normalnie rozwarte) lub rozwierne (normalnie zwarte) oraz styki bistabilne (w których obie pozycje są trwałe). Ponadto rozróżnia się włączniki/wyłączniki (mają dwa wyprowadzenia i styk łączący/rozłączający je) oraz przełączniki (mają 3 lub więcej wyprowadzeń i styk przełączający przepływ pomiędzy jednym z wyprowadzeń a pozostałymi). Stan zwarcia określa się często mianem "zamknięty" (ang. "close"), a rozwarcia mianem "otwarty" ("open"). Należy (zwłaszcza w systemach cyfrowych) pamiętać że przełączniki potrafią drgać - na ogół

zamiast pojedynczego impulsu generują całą serię. Zobacz w Sieci: zjawiska stykowe[22], gaszenie łuku (en)[23], ochrona styków (en)[24]gasik (en)[25].

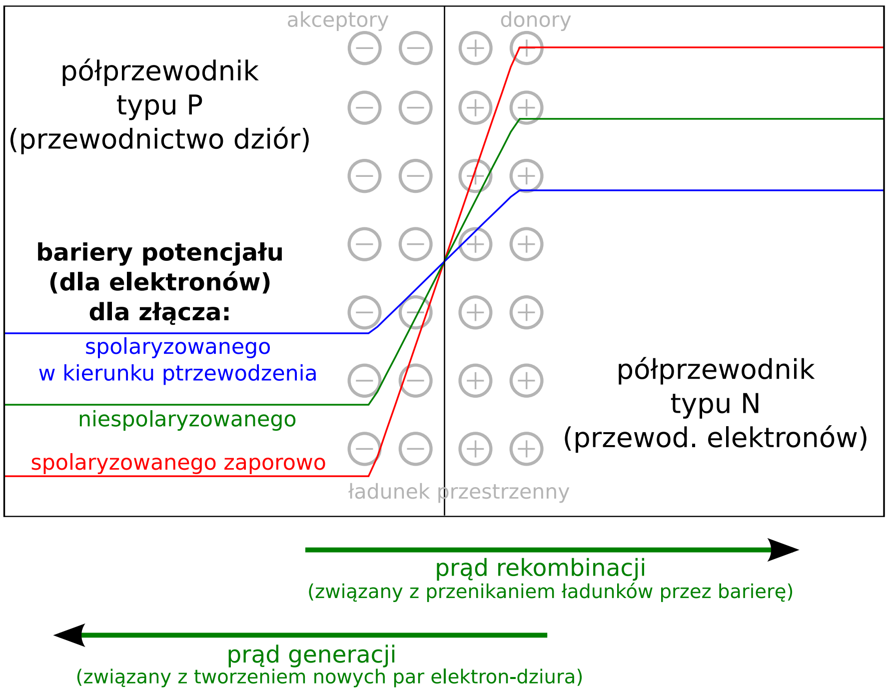
Diody

Dioda[26] jest elementem o nieliniowym oporze elektrycznym, zasadniczo przepuszczającym prąd tylko w jedną stronę. Wyróżnia się m.in. diody:

- prostownicze[27] (takie zwykłe)
- szybkie (lepsze czasy przełączania - każdą dioda charakteryzuje pewien czas przez który po zmianie polaryzacji płynie impuls prądu wstecznego, o określonej dla danej diody amplitudzie)
- Schottky'ego (przechowują niewielki ładunek, więc szybciej się przełączają)
- Zenera[28] (przewodzą także w przeciwnym kierunku, jednak po przekroczeniu ściśle określonego napięcia zaporowego)
- świecące (LED[29], emitują światło)
- fotodiody (oświetlenie spolaryzowanego zaporowo złącza powoduje wzrost prądu wstecznego)

Podstawowymi parametrami diód jest maksymalne napięcie wsteczne (określa przy jakim napięciu przyłożonym w kierunku zaporowym może nastąpić przebicie) oraz prąd przewodzenia (maksymalny prąd jaki możemy przepuścić przez element, aby go nie zniszczyć). Jest również charakterystyczne dla materiału napięcie przewodzenia określające spadek napięcia na elemencie (dla krzemu 0,5 do 0,8 V).

Dioda półprzewodnikowa[30] to w istocie pojedyńcze złącze p-n[31]. Niespolaryzowane złącze odpowiada sytuacji pokazanej na poniższym rysunku (kolor zielony). W takim wypadku prąd rekombinacji równoważony jest poprzez prąd generacji.



Polaryzacja w kierunku przewodzenia (gdy $U_P > U_N$) powoduje obniżenie bariery potencjału (dokładniej to obu barier - pokazanej na rysunku bariery dla elektronów oraz analogicznej bariery dla dziur). Efektem tego jest wzrost prądu rekombinacji, co przy niezmienionym prądzie generacji (zależy on od złącza a nie od bariery potencjału), prowadzi do powstania wypadkowego prądu płynącego przez złącze.

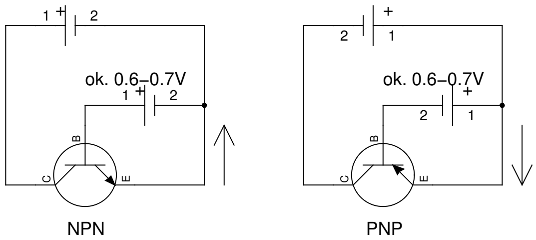
Typową eksponencjalną charakterystykę $I_F(U_F)$ złącza PN można przedstawić wzorem $I_F = I_S \exp(U_F/\varphi_T - 1)$, gzie $\varphi_T = kT/e$ i typowo wynosi około 25mV, natomiast I_S zależne jest od typu złącza i konkretnego egzemplarza. Dziki temu możemy zapisać rezystancję dynamiczną diody (związaną z spadkiem napięcia na przewodzącym złączu) jako $r_D = \varphi_T / I_{FQ}$. Tą cechę diody można wykorzystać np. do stworzenia regulowanego dzielnika (regulowane źródło prądu DC + dioda) będącego w stanie zmniejszać amplitudę sygnałów dużej częstości (gdzie nie możemy zastosować potencjometru, gdyż za bardzo zagłuszyłby sygnał).

Tranzystor

tranzystor[32] - jest to element o regulowanym elektronicznie oporze (tak na prawdę regulowany jest przepływający prąd), często wykorzystywany do wzmacniania sygnałów lub jako przełącznik elektroniczny.

Tranzystor bipolarny[33] posiada trzy wyprowadzenia - emiter (E), baza (B), kolektor (C), przepływający przez niego prąd reguluje się poprzez przyłożenie napięcia między bazą a emiterem. W tranzystorach PNP prąd płynie od emitera (o wyższym potencjale) do kolektora, w NPN na odwrót. Należy też pamiętać że tranzystor bipolarny to nie bramka logiczna czy coś w tym stylu - jeżeli przyłożymy napięcie w kierunku przewodzenia do bramki to prąd popłynie nawet gdy nie ma przyłożonego napięcia kolektor - emiter (bramka nie jest izolowana).

Przy wzmacnianiu sygnałów tranzystor pracuje w stanie aktywnym[34] czyli napięcie przyłożone do bazy jest pomiędzy napięciem kolektora a emitera. W przypadku wykorzystywania jako przełącznik tranzystor pracuje w stanach zatkania (nie przewodzi) lub nasycenia (nie ogranicza). Poniższa ilustracja przedstawia podstawową polaryzację tranzystora.



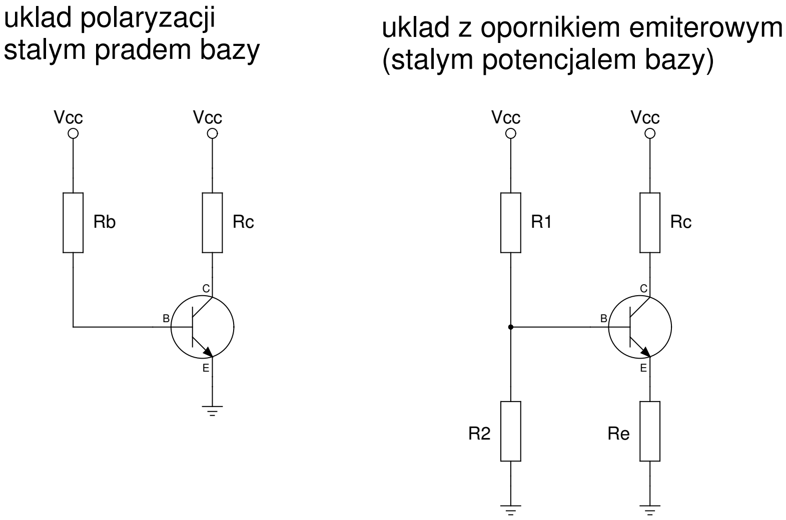
tranzystory NPN i PNP spolaryzowane w kierunku przewodzenia (strzalka pokazuje kierunek pradu)

Strzałka w symbolu tranzystora (umieszczana na emiterze) pokazuje kierunek przepływu prądu przez emiter. Pokazuje ona kierunek przepływu prądu przez złącz P-N (diodę) baza-emiter, zatem wskazuje także kierunek prądu bazy.

W stanie aktywnym prąd kolektora jest regulowany poprzez napięcie baza-emiter. Zatem na tranzystor możemy patrzeć jak na diodę w której rozdzieliliśmy styk odpowiedzialny za przyłożone napięcie od styku którym płynie prąd - tak jak w diodzie **prąd zależy wykładniczo od przyłożonego napięcia**, tyle że napięcie przykładamy pomiędzy E i B, a prąd płynie głównie pomiędzy E i C (jest także przepływ pomiędzy B i E związany z rekombinacją elektronów w bazie i wstrzykiwaniem ładunków baza-emiter - ze względu na różnice w domieszkowaniu słabszym niż wstrzykiwanie emiter-baza). Zatem prąd płynący przez tranzystor (prąd kolektora) zależy tylko od napięcia diody baza-emiter.

Ciekawym przypadkiem jest wymuszone $I_C = 0$ (gdy kontakt kolektora pozostaje niepodłączony) - wtedy prąd bazy musi wzrosnąć do takiej wartości aby zrównoważyć prąd emitera (on nie ulega zmianie gdyż zależy tylko od napięcia baza-emiter), natomiast na złączu kolektorowym będzie 0.5V (trochę mniej niż spadek na diodzie gdyż kolektor jest słabo domieszkowany).

Możliwe jest także **sterowanie tranzystora prądem bazy, a nie napięciem BE**. W takim wypadku napięcie pomiędzy bazą a emiterem wynosi około 0.6V - 0.7V (warto pamiętać o tym napięciu przy obliczaniu rezystancji wymaganej aby uzyskać pożądany prąd bazy). W wypadku tym do bazy wprowadzamy dodatkowe ładunki (w NPN są to dziury), co owocuje wciąganiem w ten obszar ładunków przeciwnych z emitera. Większość z nich jednak nie zdąży zrekombinować w bazie (jest ona cienka) tylko przeleci do kolektora, którego prąd dany jest zależnością: $I_C = \beta I_B$, gdzie β to wzmocnienie tranzystora. Zatem tranzystor zachowuje się jak (regulowane prądem) źródło prądowe. W obwodzie kolektora wprowadza się zazwyczaj rezystor (Rc) mający na celu zamianę sygnału prądowego uzyskiwanego dzięki tranzystorowi na napięcie, z którego jest łatwiej korzystać niż z prądu.

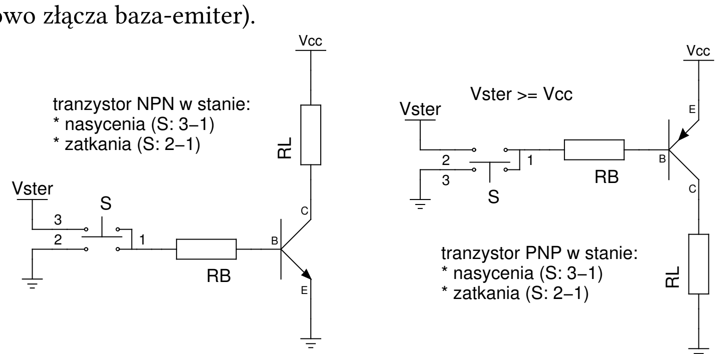


Ze względu na znaczny rozrzut wartości β pomiędzy egzemplarzami układ taki jest trudny do praktycznego stosowania (rezystor ustalający prąd bazy należałoby dobierać do konkretnego tranzystora / stosować rezystor regulowany i go dostrajać), dlatego najczęściej stosowany jest "**układ z stałym potencjałem bazy**". W układzie takim do bazy podłączamy pewien ustalony potencjał (na tle wysoki aby nieznajomość spadku na złączu PN - czy to jest 0.6V czy 0.7V itp - była pomijalna), a poprzez zastosowanie opornika emiterowego (Re) ustalamy prąd emitera. Na jego wartość nie ma wpływu to co dzieje się w kolektorze, gdyż zależy on tylko od napięcia baza-emiter (w przypadku niepodłączonego kolektora całość prądu emitera płynie przez bazę, w przeciwnym razie przez bazę płynie $I_B = \frac{I_E}{\beta+1}$). Dzięki takiemu zabiegowi (ustaleniu prądu emitera) udało się bardzo skutecznie zminimalizować wpływ wartości β na prąd kolektora gdyż $I_C = I_E - I_B = I_E \frac{\beta}{\beta+1}$, a jako że $\beta \gg 1$ to $I_C \approx I_E$.

W układzie z stałym potencjałem bazy jeżeli spadek na rezystancji wewnętrznej źródła, którym jest dzielnik R1 R2, jest istotny (w porównaniu z nieznajomością spadku napięcia na złączu baza-emiter) należy uwzględniać tą rezystancję w obliczaniu prądu emitera: $E_z - I_E \frac{1}{\beta+1} R_z - U_{BEP} - I_E R_E = 0$. Warto wspomnieć iż za polaryzację bazy może być odpowiedzialne inne źródło napięcia niż klasyczny dzielnik (np. dzielnik z diodą Zenera). Zobacz w Sieci: Jak Działa Tranzystor Bipolarny[35], Obliczanie I Ustalanie Punktu Pracy[36].

Stan nasycenia polega na polaryzacji obu złącz w stanie przewodzenia (na typie P napięcie wyższe o około 0.6V niż na typie N). Na tranzystorze w stanie nasycenia występuje spadek napięcia $U_{CE} = U_{BE} - U_{BC} \approx 0.7 - 0.5 = 0.2$V. Jeżeli spadek ten może być przeszkodą należy rozważyć zastosowanie trybu nasyconego inwersyjnego zamieniony kolektor z emiterem), gdzie on jest dużo bardziej bliski zeru. Tranzystor wchodzi w ten stan gdy $V_E < V_B > V_C$ (NPN) lub $V_E > V_B < V_C$ (PNP). W przypadku regulacji prądem bazy ma to miejsce gdy $I_C = \beta I_B > I_{C_{max\ ukl}}$ (czyli gdy spadek na obciążeniu wynikły z obliczonego prądu kolektora spowodowałby odłożenie na nim napięcia większego niż napięcie zasilania). Tranzystor w stan nasycenia wprowadzać należy z (dwu - trzy krotnym) zapasem prądu nasycającego, czyli $R_B << \frac{U_{Ster} - U_{BE}}{I_B}$.

Stan zatkania polega na zaporowej polaryzacji obu złącz czyli $V_E \geq V_B \leq V_C$ (NPN) lub $V_E \leq V_B \geq V_C$ (PNP). W tym wypadku należy szczególnie zwrócić uwagę na (niewielkie) napięcia przebicia spolaryzowanego zaporowo złącza baza-emiter).



Nasycenie NPN gdy: $I_C > V_{cc}/R_{L_{min}}$, gdzie $I_C = \beta_{min} I_B$, a $I_B = \frac{V_{ster} - 0.7V}{R_B}$ zatem otrzymaliśmy warunek na rezystor bazy: $R_B < \beta_{min}(V_{ster} - 0.7V)\frac{R_{L_{min}}}{V_{cc}}$. Dla PNP rozumujemy analogicznie, tyle że prąd bazy zależy od napięcia zasilania a nie sterowania, czyli: $R_B < \beta_{min}(V_{cc} - 0.7V)\frac{R_{L_{min}}}{V_{cc}}$. Warto zauważyć, że $\frac{R_{L_{min}}}{V_{cc}} = 1/I_{C_{max\ ukl}}$.

JFET (FET)

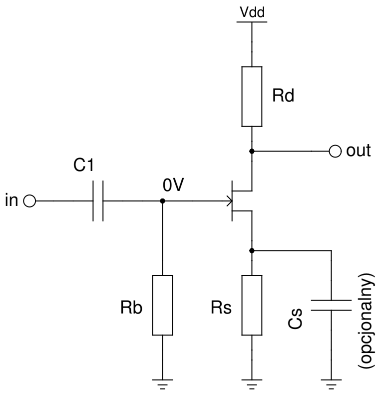
Tranzystor unipolarny[37] (polowy) posiada trzy wyprowadzenia - dren (D), bramka (G), źródło (S), regulacja odbywa się poprzez regulację napięcia między źródłem a bramką. W tranzystorach tych sterowanie odbywa się polem elektrycznym (z tąd nazwa polowy), a prąd bramki (gdy tranzystor jest dobrze spolaryzowany) jest pomijalnie mały. Dzięki temu mogą służyć do uzyskania wejścia o dużej rezystancji wejściowej.

Dostępne na rynku tranzystory JFET[38] z kanałem N steruje się poprzez ujemną polaryzację bramki wobec źródła. W przypadku gdy potencjał bramki jest odpowiednio ujemny (mniejszy od charakterystycznego - w zasadzie dla danego egzemplarza - napięcia "odcięcia") tranzystor nie przewodzi (prąd DS pomijalne mały). W przeciwnym wypadku w zależności od przyłożonego napięcia DS tranzystor ten zachowuje się jak regulowane źródło prądowe (gdy to napięcie większe od różnicy pomiędzy obecnym napięciem GS a napięciem odcięcia) lub regulowany rezystor (gdy mniejsze). Maksymalny dla danego napięcia prąd (oznaczany) płynie gdy napięcie pomiędzy bramką a źródłem jest równe zero (w zasadzie troszkę większe od zera, ale tak aby nie spolaryzować złącza w stan przewodzenia).

Niestety tranzystory te cechują się dużym rozrzutem kluczowych parametrów (napięcie GS przy którym następuje zatkanie oraz maksymalny prąd przewodzenia) pomiędzy egzemplarzami. Najprostszym przykładem zastosowania jest źródło prądowe utworzone poprzez podłączenie bramki do masy oraz źródła poprzez (regulowany) opornik do masy. Tranzystor ten w stanie z otwartym złączem kanał-bramka może być wykorzystany jako dioda mało upływowa.

$$I_D = \begin{cases} 0, & U_{GS} < U_T \\ \beta \cdot (U_{GS} - U_T)^2, & U_{GS} \geq U_T \wedge U_{DS} > U_{GS} - U_T \\ \beta \cdot U_{GS} \cdot [2(U_{GS} - U_T) - U_{DS}], & U_{GS} \geq U_T \wedge U_{DS} \leq U_{GS} - U_T \wedge U_{DS} \geq 0 \end{cases}, \text{ gdzie } \beta = \frac{I_{DSS}}{{U_T}^2}$$

Tranzystory J-FET cechują się symetrią pomiędzy źródłem a drenem, ale przy projektowaniu układów z nimi warto wiedzieć którą elektrodę traktujemy w jaki sposób. Tranzystory J-FET zazwyczaj polaryzuje się poprzez połączenie przez duży opór (R_B rzędu giga omów) bramki do masy, co wraz z małym prądem bramki zapewnia na niej potencjał 0V. Jako że J-FET przy $U_{GS} = 0$ przewodzi (i to maksymalny prąd) to taka polaryzacja powoduje przepływ prądu DS co w związku z oporem R_S pomiędzy źródłem a masą prowadzi do podniesienia potencjału na źródle czyli pojawienia się ujemnego napięcia $U_{GS} = U_{R_S}$ i ograniczenia prądu.



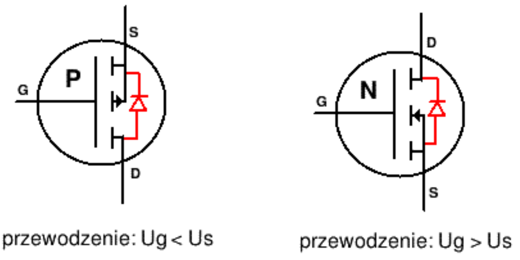
Jako że występują dwa tryby pracy JFETów to przy wykonywaniu obliczeń konieczna jest identyfikacja z którym trybem pracy mamy doczynienia. Możemy wykonać to orientacyjnie na podstawie porównania wartości R_D i R_S - gdy $R_D >> R_0$ to możemy podejrzewać obszar triodowy / nienasycenia ($I_D = \beta \cdot U_{GS} \cdot [2(U_{GS} - U_T) - U_{DS}]$), gdzie tranzystor zachowuje się jak opornik regulowany. Jeżeli jednak nie zgodzą nam się znaki napięć musimy policzyć dla obszaru pentodowego / nasycenia ($I_D = \beta \cdot (U_{GS} - U_T)^2$). Warto także pamiętać że $I_D = \frac{|U_{GS}|}{R_S}$

MOSFET (MOS)

Ważną grupą tranzystorów polowych są tranzystory z izolowaną bramką (IGFET[39]). Najczęściej wytwarzane są one w technice MOSFET[40]. Tranzystory te stanowią podstawę układów scalonych wykonywanych w technologi CMOS[41]. Dzięki izolowanej bramce tranzystory te regulowane są jeszcze bardziej napięciem a nie prądem (tu warto wspomnieć o pewnej ciekawostce - gdy elektrodę bramki wprowadzimy w stan NC (niepodłączona) tranzystor utrzymuje (przez dość długi czas) stan poprzedni). W wnętrzach układów scalonych spotyka się tranzystory z wieloma bramkami, jak również wyprowadzoną niezależnie elektrodą podłoża (normalnie zwarta do źródła).

Tranzystor z kanałem typu P ma regulowane przewodzenie w kierunku od źródła do drenu (w drugą stronę zawsze przewodzi), wprowadzenie w stan przewodzenia odbywa się poprzez podanie na bramkę napięcia mniejsze-

go od napięcia źródła. Tranzystor z kanałem typu N ma regulowane przewodzenie w kierunku od drenu do źródła (w drugą stronę zawsze przewodzi), wprowadzenie w stan przewodzenia odbywa się poprzez podanie na bramkę napięcia większego od napięcia źródła.

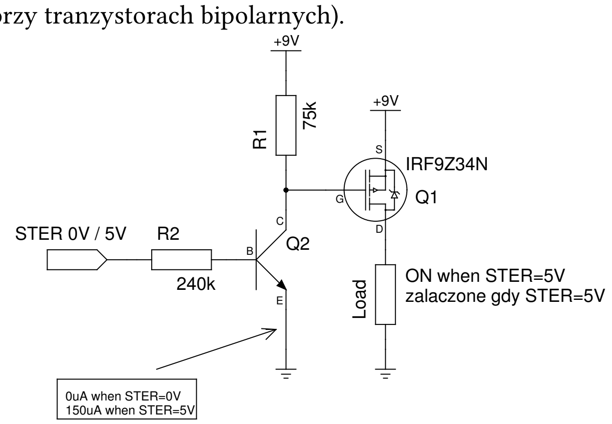


(zaznaczona na czerwono dioda niekiedy jest pomijana w oznaczeniach)

Warto wspomnieć iż występują także tranzystory bipolarne z izolowaną bramką (IGBT[42]), są one powszechnie stosowane w urządzeniach elektroenergetycznych do przełączania znacznych prądów. Na tranzystor taki możemy patrzeć jak na sterowany za pośrednictwem MOSFETa tranzystor bipolarny.

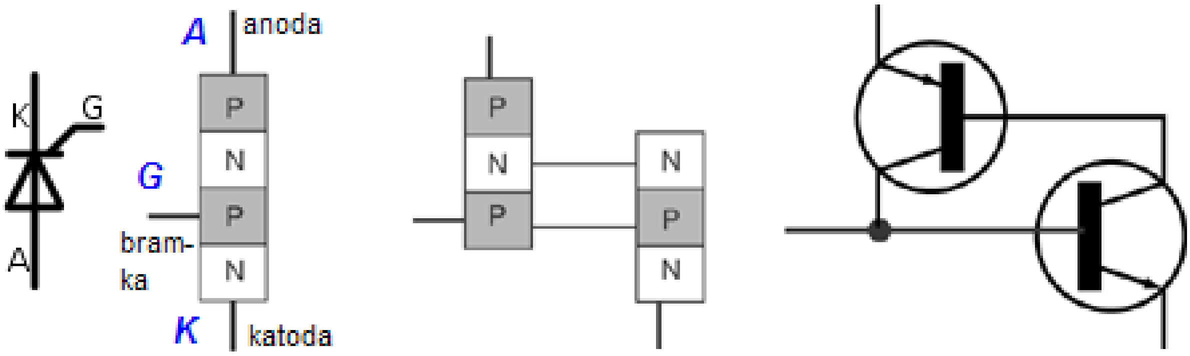
Klucz

Tradycyjny klucz na tranzystorze NPN jest bardzo dobrym rozwiązaniem gdy zadowala nas odłączanie od zasilanego obwodu potencjału niższego (masy). W przeciwnym razie powinniśmy użyć klucza PNP jednak on wymaga aby napięcie sterowania osiągało co najmniej wartość napięcia zasilania, co także często jest niezadowalające. Rozwiązaniem takiego problemu wydaje się zastosowanie układu złożonego z tranzystora PNP i tranzystora P-MOSFET, gdzie MOSFET jest sterowany z klucza NPN. Dodatkowo dzięki takiemu rozwiązaniu (przy zastosowaniu odpowiedniego MOSFETa) uzyskujemy praktycznie pomijalny spadek napięcia na takim kluczu (znacznie mniejszy niż umowne 0.2V przy tranzystorach bipolarnych).



Tyrystor, triak

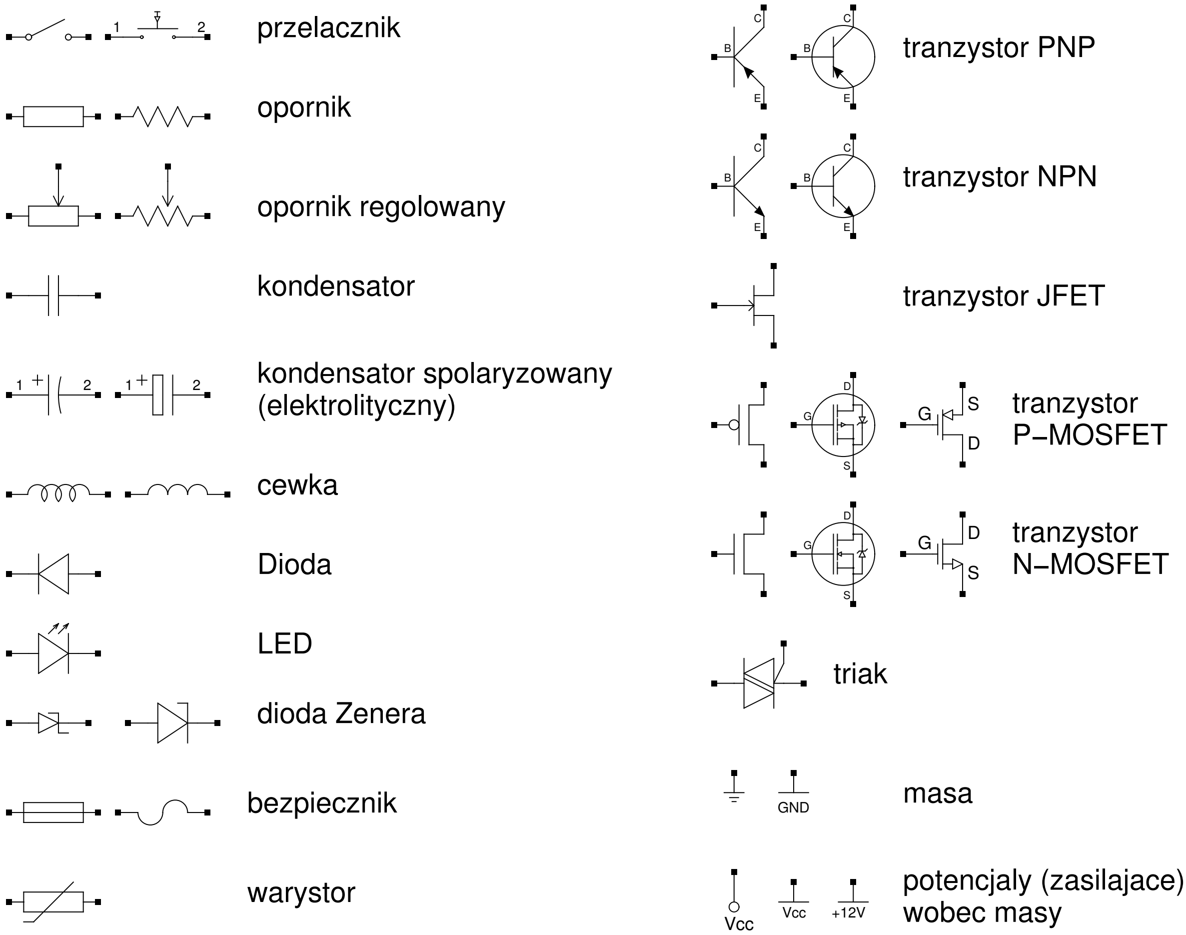
tyrystor[43] - jest to element o regulowanym elektrycznie stanie przewodzenia, przewodzić on może od anody do katody (tylko w tą stronę), pod warunkiem że zostanie wyzwolony impulsem bramki (dodatnie napięcie względem katody) bądź wzrostem napięcia przyłożonego. W odróżnieniu od tranzystora tyrystor przewodzi również po zaniku napięcia przyłożonego do bramki (przerywa dopiero gdy zostanie przerwane przewodzenie). Triak[44] jest w zasadzie dwukierunkową wersją tyrystora odpowiadającą funkcjonalnie połączonym antyrównolegle dwóm tyrystorm. W zrozumieniu jak to działa przydany może być schemat zastępczy tyrystora na tranzystorach bipolarnych:



Inne elementy (analogowe), symbole

Oprócz omówionych mamy jeszcze wiele innych elementów półprzewodnikowych w tym układów scalonych. Wspomnę może tutaj o jednym z nich - NE555 jest to bardzo popularny układ czasowy służący do generacji drgań i wytwarzania impulsów. Dokumentacje do różnych układów elektronicznych (tzw. datasheet) można znaleźć na stronie producenta układu lub na stronach poświęconym notom katalogowym układów elektronicznych, takich jak EleNota[45].

Każdy element ma swój symbol graficzny (a nawet kilka ...). Poniżej zamieszczam obrazek (PNG) prezentujący najpopularniejsze symbole podstawowych elementów elektronicznych (symbole1.sch[46]).



Radiatory

Często elementy elektroniczne wymagają dodatkowego chłodzenia - przynajmniej w postaci radiatora. Jego dobór przeprowadza się następująco:

```
{rezystancja termiczna radiatora} =   ( {temperatura maksymalna układu} - {temperatura
↪  otoczenia} ) / { moc wydzielana }
                                    - ( {rezystancja termiczna złacze-obudowa} +
↪  {rezystancja termiczna obudowa-radiator} )
```

Gdzie `{temperatura maksymalna układu}` i `{rezystancja termiczna złacze-obudowa}` to dane katalogowe chłodzonego układu (tranzystora, triaka, ...), zaś jako `{rezystancja termiczna obudowa-radiator}` przyjmuje się wartość od 1K/W (bezpośrednie przykręcenie) do 0.2K/W (pasta silikonowa).

Dla przykładudla BTA16, temperatury otoczenia 50 stopni C i 10 W strat (co odpowiada przepuszczaniu przez triaka 10A) mamy: `{rezystancja termiczna radiatora przy 10 W strat} = (125-50) / 10 - 2.1 - 0.1 = 5.3K/W` czyli wartość rezystancji termicznej radiatora musi być **mniejsza** lub równa od 5.3K/W.

Zobacz także w Sieci: artykuł w EdW[47]

Warte uwagi układy scalone

Wybrane układy serii 74HC:

- bramki
  - NOT
    - 04[48]
  - AND
    - 2 wejściowa - 08[49]
  - OR
    - 2 wejściowa - 32[50]
  - XOR
    - 2 wejściowa - 86[51]
  - NAND
    - 2 wejściowy - 00[52]
    - 4 wejściowy - 20[53]
    - 8 wejściowy - 30[54]
    - 2 wejściowy (otwarty kolektor) - 03[55]
    - 2 wejściowy (Shmitt) - 132[56]
  - NOR
    - 2 wejściowa - 02[57]
  - XNOR
    - 2 wejściowa - 266[58]
- przerzutniki (FF)
  - 2x JK - 112[59]
  - 4x D - 175[60]
  - 8x D - 273[61]
  - 8x D clock enable - 377[62]
  - **8x D 3-state out** - 574[63]
  - 8x D inv 3-state out - 564[64]
- bufory (BUF)
  - 4x 3-state buffer - 125[65], 126[66]
  - 6x Schmitt Trigger Buffer - 14[67]
  - **8x 3-state buffer bi-directional** - 245[68]
- dekodery, demultiplexery (DECODER, DEMUX)
  - 2 na 4 - 139[69]
  - **3 na 8** - 138[70]
  - 4 na 10 (BCD) - 42[71]

- multipleksery (MUX)
  - 2 wejściowy - 257[72]
  - 4 wejściowy - 153[73]
  - 8 wejściowy - 251[74]
- ENCODERy
  - 8 na 3 - 148[75]
  - 10 (BCD) na 4 - 149[76]
- rejestry przesówne (SHIFT_REG)
  - PISO - 165[77]
  - SIPO - 164[78]
  - left-right - 194[79]
- liczniki (COUNTER)
  - 2x 4bit - 393[80]
  - 14bit - 4020[81]
  - 12bit - 4040[82]
  - 4bit z presetem - 193[83]
- inne
  - 4x analog switch - 4066[84]
  - analog (de)mux - 4051[85], 4052[86], 4053[87]
  - 8x latch 3-state out - 373[88], 573[89]
  - komparator liczb 4 bit - 85[90]
  - monostable multivibrator - 123[91]

  Zobacz więcej: Seria 7400[92], wykaz układów serii 4000[93], wykaz układów serii 7400[94].

Inne układy:

- mikrokontrolery AVR
  - Atmega8
  - Atmega16
  - Atmega164, Atmega324, Atmega644
- drivery interfejsów
  - USB to 8bit - FT245
  - USB to UART - FT232
  - USB to 4xUART (w tym dwa uniwersalne - mogące działać jako I2C, SPI, JTAG - FT4232
  - RS485 - MAX485[95]/SN75176B
  - RS232 - MAX232[96]/ST232
  - Ethernet na SPI - ENC28J60
- układy na szynę I2C
  - 8bit IO z zaawansowanym kontrolerem przerwań (niestety INTF zdaje się nie działać tak jak opisano w dokumentacji - łapie tylko pierwsze przerwanie) - MCP23008[97]
  - open-drain 8bit IO z kontrolerem przerwań - MCP23009[98]
  - 8bit IO z prostym odczytem (mniejsze obciążenie magistrali, dzięki dwóm dostępnym wariantom adresu zaszytego w krzemie do 16 układów na szynie) - PCF8574A(P)[99]
  - 4 analog input, 1 analog output - PCF8591[100]
  - multipleksery magistrali I2C - PCA9540[101], PCA9542[102], PCA9544[103], PCA9548[104]
- transoptory
  - cyfrowe - 6N136[105], 6N137[106]
  - analogowe - 4N25[107], 4N35[108]
  - optotriaki - MOC3023[109], MOC3063[110], itp
- elementy załączające
  - tranzystory NPN - BC337[111]
  - tranzystory PNP - BC327[112]
  - tranzystory N-MOSFET - IRFZ44N[113], BS170[114]
  - tranzystory P-MOSFET - IRF9Z34N[115]
  - tranzystory JFET - BF245[116]
  - triaki - BTA16[117]
- inne
  - wyświetlacze LCD z sterownikiem HD44780[118]

— termometr 1-wire - DS18B20[119]
— układ czasowy - NE555[120]
— wzmacniaz operacyjny - LM358[121], TL081[122]
— komparator - LM239N[123]

Warte zainteresowania mogą być także klasyczne kości architektury PC takie jak 8255[124], 8259[125], itp, a także popularne współczesne mikrokontrolery z rodzin takich jak AVR, PIC, ...

Linki

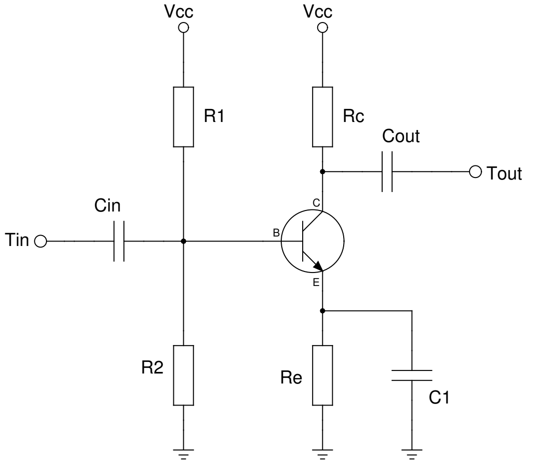
Zachęcam też do zapoznania się z częścią ”wstępu do techniki” którą poświęciłem elektronice, omawiam tam m.in. prosty układ elektroniczny obrazujący najistotniejsze aspekty tej dziedziny oraz moimi projektami: układ ograniczający prąd rozruchowy transformatora, układu konwersji pomiędzy różnymi standardami UART (RS232, RS485, RS_TTL), niestabilizowanego zasilacza buforowego, konwertera poziomów TTL, konwerter pomiędzy USB a UART (TTL).

Zobacz w Sieci: Wprowadzenie do elektroniki[126], Lessons In Electric Circuits[127], Lessons In Industrial Instrumentation[128], Elektronika (nie tylko) analogowa[129], Elektronika Wirtualna[130], Elportal - Kopalnia skarbów[131], Elektronika Praktyczna - archiwum[132], Laboratorium z techniki cyfrowej w AGH[133], Pracownia elektroniczna FUW - podstawy elektroniki[134], Dlaczego neonówka świeci?[135], Bezprzewodowy wykrywacz napięcia czyli ”Jak znaleźć usterkę w lampkach choinkowych?”[136], Przystawka do pomiaru indukcyjności (kit AVT-139)[137], Obudowy układów scalonych[138], Elektroda.pl[139], Mirley - Elektronika i Programowanie[140], Novena (open laptop)[141],
Signals and Systems[142], Microprocessor Design[143], Control Systems[144], Embedded Systems[145], Robotics[146], How To Assemble A Desktop PC[147], wyświetlacz[148], wyświetlacz ciekłokrystaliczny[149], OLED[150], Plastic leaded chip carrier[151], Digital Electronics[152], Electronics Handbook[153], Analogue Electronics[154], Learn Electronics[155], Electronics Communication[156], Electronics Fundamentals[157], Fundamental Digital Electronics[158], Digital Circuits[159], Embedded Control Systems Design[160], Systems Theory[161], Analog and Digital Conversion[162], Electric Motors And Generators[163], Semiconductors[164], Signal Processing[165], Engineering Tables[166], Basic Electrical Generation and Distribution[167], Kategoria:Elektronika[168], Kategoria:Elektroakustyka[169], Kategoria:Optoelektronika[170], Control rooms[171], Electronics[172], Computers[173].

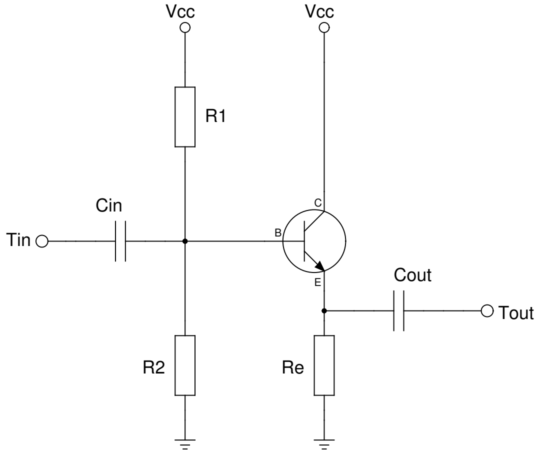
Układy tranzystorowe

Wzmacniacz małych sygnałów

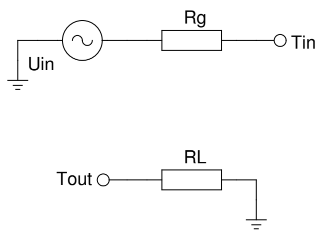
Typowym zastosowaniem tranzystora jest wzmacnianie sygnałów. Dalej będziemy przez jakiś czas przyjmować że mamy doczynienia z ”małym sygnałem” czyli takim którego dołożenie powoduje niewielkie odchylenia od punktu pracy (takie że można zaniedbać krzywiznę charakterystyki). Wyróżnić można 3 podstawowe układy pracy tranzystora bipolarnego w wzmacniaczu sygnału (pokazane na przykładzie układu polaryzacji z opornikiem emiterowym):

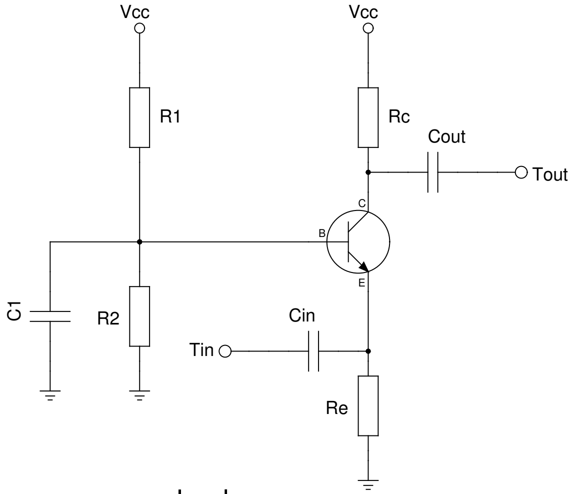


wspolny emiter



wspolny kolektor





wspolna baza

W przypadku pracy w układzie wspólnego emitera, generator sygnału dokładamy poprzez kondensator do spolaryzowanej w stanie aktywnym bazy. W efekcie tego na stały prąd wyjściowy nakłada się prąd związany z sygnałem wejściowym wynoszący $I_C|_{AC} = g_m U_{b'e}|_{AC}$, gdzie $g_m = \frac{I_{CQ}}{\varphi_T} \approx \frac{1}{r_{eb'}}$ jest nachyleniem charakterystyki w punkcie pracy, a $U_{b'e}|_{AC}$ jest sygnałem dołożonym na złącze baza-emiter (w przybliżeniu równym sygnałowi wejściowemu).

W prezentowanej sytuacji (układ z opornikiem emiterowym) przy rozważaniu mechanizmu tego procesu lepiej byłoby rozumować właśnie poprzez rezystancję dynamiczną emitera $r_{eb'}$ (a nie g_m, które jest właściwe dla polaryzacji stałym prądem bazy), gdyż w układzie takim zmiana napięcia zasilającego przekłada się w niezmiennym stosunku (określonym głównie przez wartość oporu emiterowego) na prąd emitera, a ten jest w przybliżeniu równy prądowi kolektora. Zatem uzyskanie wzmocnienia innego niż 1 wymaga zwarcia emitera dla sygnałów zmiennych do masy (kondensator CE) i wtedy właśnie opór dynamiczny $r_{eb'}$ powoduje że uzyskujemy skończone wzmocnienie.

Każdy wzmacniacz charakteryzuje się następującymi parametrami (podane sposoby obliczania dla układu pracy wspólny emiter):

- rezystancja wejściowa - utworzona przez rezystancję zastępczą oporników ustalających punkt pracy oraz oporu dynamicznego złącza PN pomiędzy bazą a emiterem wynoszącą: $r_{b'e} = \beta r_{eb'} = \beta \frac{\varphi_T}{I_E} \approx \beta \frac{\varphi_T}{I_C}$
- rezystancja wyjściowa - utworzona przez rezystor łączący kolektor z zasilaniem
- wzmocnienie napięciowe zwykłe $|k_u| = U_{wy}/U_{we} = g_m R_C$ (niekiedy umawiamy się że obejmuje też rezystancję obciążenia a nie tylko R_C)
- wzmocnienie napięciowe skuteczne (czyli uwzględniające rezystancję wewnętrzną źródła sygnału) $|k_{us}| = k_u \frac{R_{we}}{R_{we}+R_G}$

Zobacz także: Model Hybrid-Pi[174]

OBLICZANIE UKŁADU TYPU WSPÓLNY EMITER

Ustalanie punktu pracy (bipolarny, stały prąd bazy):

1. $I_B = I_{CQ}/\beta$
2. $R_B = \frac{U_{CC}-U_{BE}}{I_B}$
3. $R_C = \frac{U_{CC}-U_{CEQ}}{I_{CQ}}$
4. rozrzut β załatwiamy licząc dla średniej i sprawdzając czy dla maksymalnej nie prowadzi do nasycenia

Obliczenie punktu pracy i parametrów układu (bipolarny, stały prąd bazy):

1. $U_{RB} = U_{CC} - U_{BE}$
2. $I_B = \frac{U_{RB}}{R_B}$
3. {\bf $I_C = \beta I_B$}
4. $U_{CE} = U_{CC} - R_C I_C$
5. $R_{WE} = R_B || r_{b'e}$
6. $R_{WY} = R_C$
7. $k_u = \frac{U_{WY}}{U_{WE}} = g_m(R_C||R_O) = \frac{I_C}{\varphi_T} \frac{R_C R_O}{R_C + R_O}$
8. $k_{us} = k_u \frac{R_{WE}}{R_{WE}+R_G}$
9. $k_i = k_u \frac{R_{WE}}{R_0}$
10. $k_{is} = \frac{I_O}{I_G} = \frac{U_O/R_O}{U_G/R_G} = k_u S \frac{R_G}{R_0}$
11. $k_{ps} = 4 k_{is} k_{us}$

Ustalanie punktu pracy (bipolarny, opornik emiterowy):

1. przyjmujemy założenie o spadku napięcia na U_{RE}
2. $R_E = \frac{U_{RE}}{I_{CQ}}$
3. $U_{RC} = U_{CC} - U_{CEQ} - U_{RE}$
4. $R_C = \frac{U_{RC}}{I_{CQ}}$
5. dzielnik - napięcie wyjściowe $U_B = U_{RE} + U_{BE}$
6. obliczamy dla sztywnego dzielnika (spadek napięcia wyjściowego związany z prądem bazy < 0.1V)
7. weryfikacja dzielnika - obliczenie U_{BE} z $E_{BZ} = I_B R_{BZ} + U_{BE} + I_E R_E$, gdzie $E_{BZ} = U_{CC} \frac{R_{B2}}{R_{B1}+R_{B2}} \approx 2.7$V
8. warto sprawdzić także czy dzielnik nie jest zbyt sztywny (nie wybija sygnału)

Obliczenie punktu pracy i parametrów układu (bipolarny, opornik emiterowy):

1. $R_{BZ} = R_{B1} || R_{B2}$
2. $E_{BZ} = U_{CC} \frac{R_{B2}}{R_{B1}+R_{B2}}$
3. $U_{RE} = E_{BZ} - U_{BE}$ (zakładamy tutaj sztywny dzielnik - spadek na R_{BZ} pomijalnie mały)
4. {\bf $I_C \approx I_E = \frac{U_{RE}}{R_E}$}
5. znając $I_B = \frac{I_E}{\beta+1}$ weryfikujemy założenie o sztywności dzielnika licząc spadek napięcia na R_{BZ} - $U_{RBZ} = R_{BZ} I_B$
6. $U_C = U_{CC} - U_{RC}$
7. $U_{CE} = U_C - U_E$
8. $R_{WE} = R_{BZ} || (r_{b'e} + \beta R_E)$ lub $R_{WE} = R_{BZ} || r_{b'e}$ (gdy mamy C_E)
9. pozostałe parametry (R_{WY}, k_u, k_{us}, k_i, k_{is}, k_{ps}) obliczamy tak samo jak przy "stałym prądzie bazy"

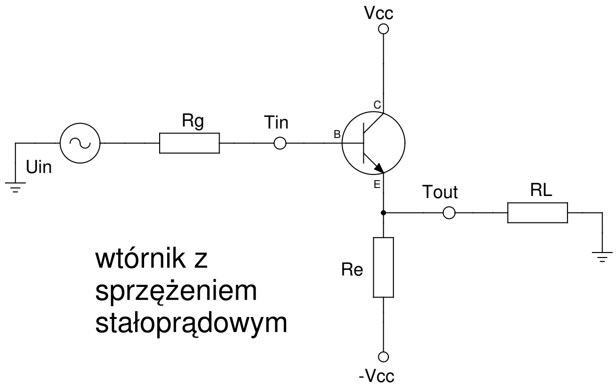
Przy obliczaniu pojemności uwzględniamy ich otoczenie rezystorowe - $R_{WE} + R_G$ dla kondensatora wejściowego, $R_{WY} + R_O$ dla kondensatora wyjściowego oraz $R_E || \left(\frac{R_G || R_{BZ}}{\beta_{AC}+1} + r_{eb'} \right)$ dla kondensatora emiterowego.

Ustalanie punktu pracy i parametrów układu (JFET):

1. $\beta = \frac{I_{DSS}}{U_T^2}$
2. obliczamy U_{GS} w oparciu o zależności: $\beta(U_{GS} - U_T)^2 = -U_{GS}/R_S$ (przyjmujemy założenie o zakresie pentodowym)
3. $I_D = -U_{GS}/R_S$
4. $U_{RD} = I_D R_D$
5. $U_{RS} = I_D R_S$
6. $U_{DS} = V_{DD} - U_{RD} - U_{RS}$
7. sprawdzamy czy zakres pentodowy - $U_{GS} \geq U_T$ i $U_{DS} \geq U_{GS} - U_T$
8. $R_{WY} = R_D$
9. $R_{WE} = R_B$
10. $k_u = -g_m R_L$ (tak samo jak w bipolarnym, ale inne $g_m = \frac{dI_D}{dU_{GS}}$ - dla zakresu pentodowego $g_m = 2\beta(U_{GS} - U_T)$)

11. pozostałe parametry (k_{us}, k_i, k_{is}, k_{ps}) obliczamy w oparciu o powyższe R_{WE}, R_{WY}, k_u, tak samo jak przy bipolarnym

Wtórnik emiteorowy



Każdy wtórnik jest układem typu "wspólny kolektor" (ale na odwrót nie zawsze). Układ taki (w odróżnieniu od wspólnego emitera czy też wspólnej bazy) jest w stanie wzmacniać nie tylko małe ale także duże sygnały. Jednak potrafi on wzmacniać tylko prąd - napięcie wyjściowe jest niemalże równe napięciu wejściowemu (dokładniej jest pomniejszone o spadek na aktywnym złączu PN).

Często wykorzystywany jest układ wtórnika z tzw. "sprzężeniem stałoprądowym" (bez odcinania składowej stałej na wejściu) i podwójnym (dodatnim i ujemnym) zasilaniem (służącym do ustalenia punktu pracy zamiast dzielnika podłączanego do bazy) przedstawiony na poniższej ilustracji.

Można spotkać się także z połączeniem dwóch tranzystorów pracujących w układzie w wtórnika. W szczególności może być to J-FET z bipolarnym (aby uzyskać dużą rezystancję wejściową i małą wyjściową), lub tranzystory tego samego typu aby uzyskać większą wypadkową wartość β (jest ona wtedy iloczynem wartości dla poszczególnych tranzystorów).

Obliczanie układu typu wtórnik emiterowy

Punkt pracy ustalany tak samo jak dla układów WE.

Ustalanie parametrów układu (bipolarny):

1. $R_{WY} = R_E || \frac{R_G + r_{b'e}}{\beta + 1}$
2. $R_{WE} = R_{b'e} + (\beta + 1)(R_E || R_L)$
3. $k_u = \frac{(\beta+1)(R_E||R_L)}{R_{b'e} + (\beta+1)(R_E||R_L)}$

Ustalanie parametrów układu (JFET):

1. $R_{WY} = R_S || \frac{1}{g_m}$
2. $R_{WE} = R_B$
3. $k_u = \frac{g_m(R_S||R_O)}{1 + g_m(R_S||R_O)}$

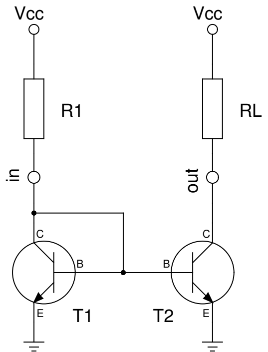
Efekt Millera

Efekt Millera[175] występuje w układach odwracających fazę i polega na tym iż impedancja widoczna z zewnątrz jest $k+1$ razy mniejsza (gdzie k - wzmocnienie układu) niż rzeczywista impedancja pomiędzy wejściem a wyjściem. Ponadto w tranzystorze występują pojemności pomiędzy bazą a emiterem oraz bazą a kolektorem. A jedna z nich w układach odwracających fazę ulega efektowi Millera. To odgrywa on istotną rolę w ograniczeniu górnej częstotliwości układów wzmacniaczy. $C_{b'e} = \frac{1}{2\pi r_{eb'} f_T} - C_{bk}$ Podawany w katalogach parametr f_T jest tylko wynikiem specyficznego pomiaru tej właśnie pojemności, a to jakie będzie związane z nią ograniczenie częstotliwości wynika z konkretnego układu w którym stosowany jest tranzystor.

Boot Strap

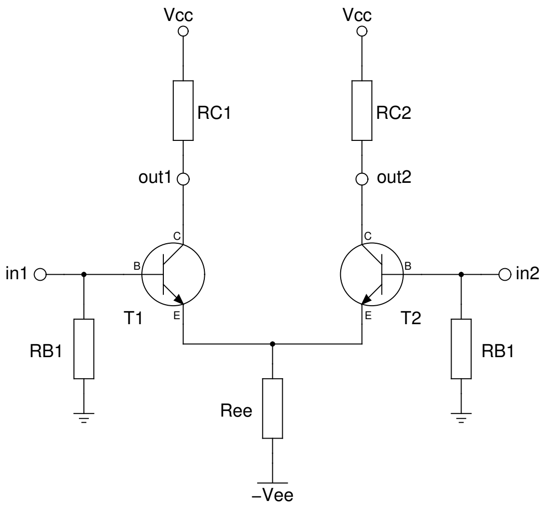
jest to efekt niejako przeciwny do efektu Millera - występuje on gdy mamy wzmocnienie napięciowe nie większe od jedności. Pozwala ono na zmniejszanie impedancji włączonej równolegle z układem wzmacniającym (w przypadku wzmocnienia równego 1 impedancja taka znika całkowicie gdyż niejako obok niej wejście jest powielane na wyjście (napięcie na niej wynosi zero. Może także robić za źródło prądowe gdy w szereg z wzmacniaczem mamy przesuwnik potencjału to na tej impedancji odkłada się stała różnica potencjałów, a zatem płynie stały prąd.

Lustro prądowe



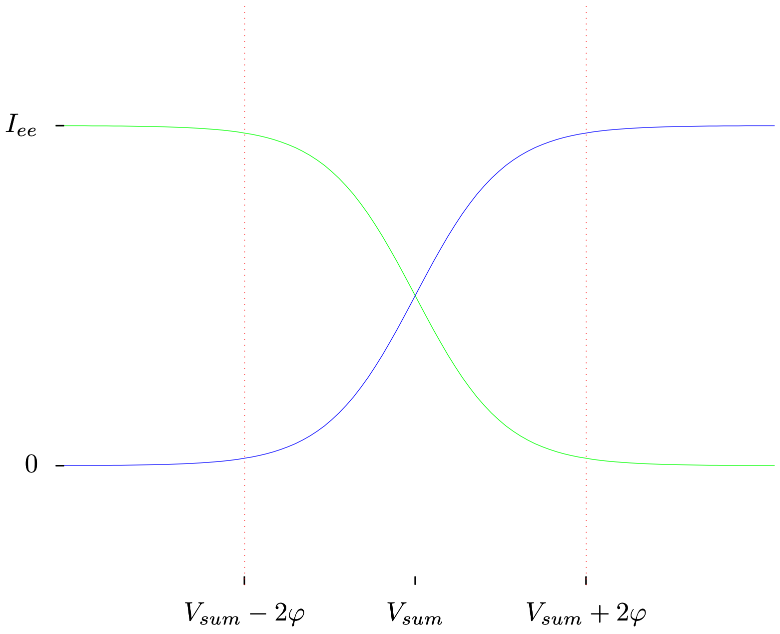
T1 i T2 muszą mieć zapewnioną tą samą temperaturę (najlepiej być wykonane w ramach jednego układu scalonego. T1 pracuje jako dioda (stosujemy tranzystor aby zapewnić taką samą charakterystykę jak T2), która wraz z R1 tworzy sprzężony termicznie z T2 dzielnik sterujący tranzystorem T2. Układ ten sterowany prądem płynącym przez R1 powoduje przepływ (niemalże) takiego samego prądu poprzez RL.

Wzmacniacz różnicowy



Układ ten może pełnić rolę klucza prądowego (gdy podany sygnał na tyle duży aby zatkać jeden z tranzystorów) lub wzmacniacza (gdy pracować będą oba tranzystory. Układ jest czuły na różnicę napięć przyłożonych do in1 i in2. Ree pełni rolę źródła prądowego (i powinien być tak dobrany wraz z -Vee aby zapewnić jego stabilność ...).

W przypadku układu z tranzystorami NPN "bardziej" przewodzić będzie tranzystor na wejściu którego jest większe napięcie (niż na tym drugim). Fakt tego że on przewodzi powoduje ustalenie się potencjału w węźle łączącym emitery obu tranzystorów na wartość napięcia jego bazy pomniejszonego o spadek na przewodzącym złączu PN. W sytuacji gdy napięcia baz obu tranzystorów są odpowiednio bliskie przewodzić będą oba, zakres ten określa się strefą przejściową i wynosi ona $4\varphi_T$.



Jeżeli pracujemy w zakresie strefy przejściowej układ ten traktować należy jako wzmacniacz (poza tym zakresem jak wspomniano jest kluczem prądowym). Posiada on wtedy wzmocnienie $k_{u_i} = \frac{R_{C_i}}{r_{eb'_1} + r_{eb'_2}}$, gdzie i jest numerem wyjścia/rezystora kolektorowego z którego pobieramy napięcie wyjściowe. Wzmocnienie wyjścia różnicowego jest sumą wzmocnień poszczególnych wyjść. Wzmacniacz ten posiada także niestety wzmocnienie sumacyjne (wzmocnienie napięcia wchodzącego równocześnie na oba wejścia). Wynika to z niemożności zastosowania idealnego źródła prądowego, a przy stosowaniu rzeczywistych zmiana napięcia na źródle (wynikła z z miany potencjału na emiterach) prowadzi do zmiany prądu płynącego przez źródło.

Jako że przy tranzystorach bipolarnych musimy umożliwić przepływ prądu bazy, a nawet przy MOSFETach musimy zapewnić polaryzację bramki określonym potencjałem, w przypadku wprowadzania na wejście sygnału z odcięciem składowej stałej (poprzez kondensator) musimy zastosować {\bf rezystory R_B} pomiędzy każdym z wejść a masą. Oba te rezystory powinny mieć tą samą wartość, gdyż inna sytuacja byłaby równoważna dołożeniu na jedno z wejść jakiegoś stałego potencjału (innego niż dołożony do drugiego wejścia) a tego nie chcemy. Podobnie gdy stosujemy sprzężenie stało-prądowe, a źródło sygnału ma znaczącą rezystancję wewnętrzną to do drugie wejście należy połączyć z masą przez taką samą rezystancję (rezystancje zastępcze widziane przez sygnał, podłączone do obu wejść powinny być jednakowe).

Często można się spotkać z modyfikacjami tego układu polegającymi na dołączaniu pomiędzy Ree a emiterami rezystorów celem powiększenia strefy przejściowej i jej linearyzacji. Możliwe jest też także rozdzielenie gałęzi emiterowej na dwie i sprzężenie ich przy pomocy kondensatora - wtedy przenoszone są tylko sygnały zmienne (każdy z emiterów ma do polaryzacji swój własny Ree) dzięki czemu można układ ten realizować na tranzystorach dyskretnych. Popularną modyfikacją tego układu jest dołączenie w miejsce RB1 i RB2 lustra prądowego przenoszącego prąd z gałęzi T1 (nie mamy wtedy wyjścia out1) na gałąź T2 (uzyskujemy wtedy dwukrotnie większe wzmocnienie na out2). Aby układ taki działał konieczne jest podłączenie rezystancji obciążenia do potencjału pomiędzy zasilaniem lustra a sumacyjnym napięciem wejściowym (składową będącą identyczną na obu wejściach).

OBLICZANIE PUNKTU PRACY, REZYSTANCJI I WZMOCNIEŃ

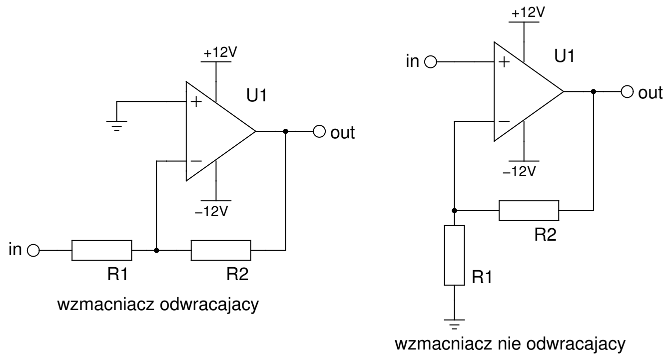
1. $I_{CQ} = I_{EE}/2$ (także w niesymetrycznym!)
2. $R_{WE} = 2r_{b'e}$
3. $R_{WY_i} = R_{C_i}$
4. $k_{u_i} = \frac{R_{C_i}}{r_{eb'_1} + r_{eb'_2}}$ (gdy korzystamy tylko z we1 to wy1 odwracające więc k_{u_1} może być podawane jako ujemne)
5. $k_{u_{sum}} = |k_{u_1}| + |k_{u_2}|$

Ewentualna niesymetryczność wpływa na zakres napięć wyjściowych.

Wzmacniacz operacyjny

Jest to układ służący do wzmacniania bardzo wiele razy różnicy napięć wejściowych. Na stopniu wejściowym posiada on wzmacniacz różnicowy z lustrem, dalej jest układ wzmacniający o dużym wzmocnieniu, a na stopniu wyjściowym wtórnik emiterowy.

Układ ten może być wykorzystywany jako komparator do porównywania napięć. Ponadto komparatory scalone posiadają taki sam symbol, jednak są innymi układami - dużo szybciej przełączają swoje wyjście przy zmianie wejścia niż robi to wzmacniacz operacyjny.



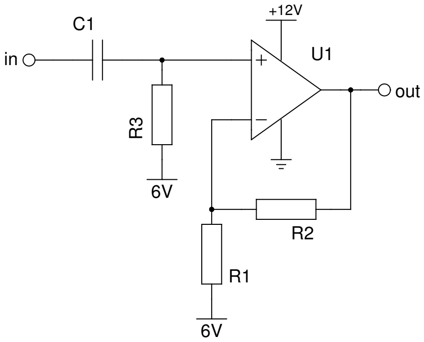
Jako iż wzmocnienie wzmacniaczy operacyjnych jest bardzo duże nie da się w praktyce wykorzystać ich bezpośrednio do wzmacniania sygnału. Robi się to z wykorzystaniem sprzężenia zwrotnego, czyli podaniem przeskalowanego sygnału wyjściowego na wejście. Dzięki temu że korzystamy z ujemnego sprzężenia układ zachowuje się stabilnie dążąc do utrzymania różnicy napięć wejściowych bliskiej zeru. Wzmocnienie wzmacniacza odwracającego fazę wynosi $k_u = -\frac{R_2}{R_1}$, a wzmacniacza nie odwracającego fazy $k_u = 1 + \frac{R_2}{R_1}$ (aby uzyskać wtórnik wystarczy zastąpić te rezystory zwarciem). Jak widać kluczowy jest tu dzielnik przez który skalujemy przekazywanie wyjścia na wejście odwracające. Wykorzystanie ujemnego sprzężenia zwrotnego owocuje także prawie zerową rezystancją wyjściową. W wzmacniaczu odwracającym $R_{we} = R_1$, natomiast w odwracającym widzimy tylko rezystancję sumacyjną wzmacniacza różnicowego (różnicowej praktycznie nie widać bo napięcie na niej bliskie zeru). Częstotliwość graniczną oblicza się z zależności $f_g = \frac{f_T}{1+\frac{R_2}{R_1}}$. Niekiedy stosuje się także tzw "R3" podłączony do nieodwracającego wejścia wzmacniacza, ma to na celu minimalizację wpływu prądów wejściowych (nie wpływa na napięcie niesymetryczności, które jest cechą samego wzmacniacza. $R_3 = R_1||R_2$ gdy sprzężenie DC, lub $R_3 = R_2$ gdy sprzężenie AC (źródło sygnału oddzielone kondensatorem).

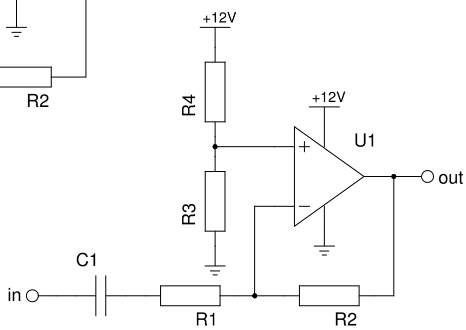
Problem nadmiaru fazy i kompensacja

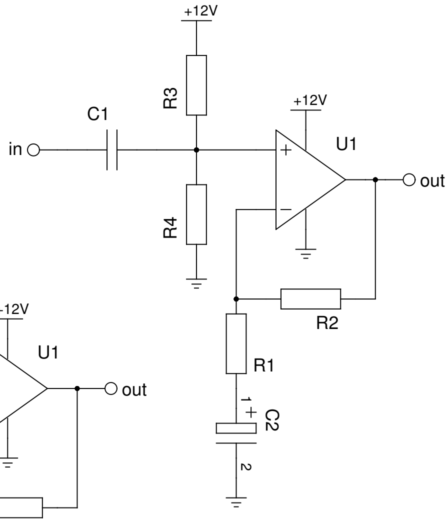
Jako iż układy te stosujemy z ujemnym sprzężeniem zwrotnym / sygnał wyjściowy podajemy na wejście odwracające fazę to należy rozwiązać problem przesunięcia fazowego sygnału wyjściowego w stosunku co do wejściowego (powinno być mniejsze od $135°$), aby nie doprowadzić do wzbudzania się układu. Uzyskuje się to poprzez zapewnienie dla częstotliwości przy których było9by takie lub większe przesunięcie fazowe wzmocnienia mniejszego od jedności. Niestety stosowany w tym celu zabieg przesunięcia niższej częstotliwości granicznej (są dwie bo dwa stopnie wzmacniające) w dół poprzez stosowanie millerowskiej pojemności mocno ogranicza pasmo przepustowe (psuje parametr f_T) wzmacniaczy operacyjnych (nie przesuwa się drugiej ze względu na wielkość potrzebnych pojemności). Poza tym konieczność ładowania/rozładowywania tej pojemności prowadzi do drugiego ograniczenia częstościowego, tym razem związanego z maksymalną prędkością zmiany sygnału (tzw. "slew rate").

Problem zasilania jednobiegunowego

W przypadku konieczności zasilenia wzmacniacza operacyjnego jedno biegunowo, należy zadbać o dodanie odpowiedniej wartości stałej do sygnału wejściowego na wzmacniacz oraz takiej samej wartości do sygnału podawanego na drugie wejście wzmacniacza. Poniżej pokazane zostały rozwiązania układowe umożliwiające realizację tego wymogu (metodę prostego uzyskania "masy wirtualnej" dla takich rozwiązań (w tym wypadku 6V) przedstawiono poniżej):







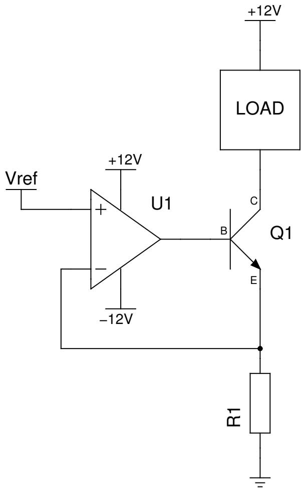
Źródła napięcia i prądu

Tranzystorowe źródło prądowe

Jednym z najprostszych źródeł prądowych jest tranzystor w 4 opornikowym układzie polaryzacji, gdzie obciążenie włączane jest w miejsce R_C. W niektórych przypadkach uzasadnione jest skorzystanie z lustra prądowego, jako źródła prądowego.

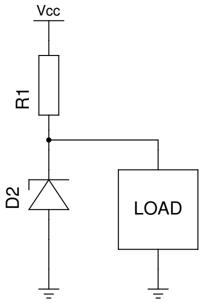
Precyzyjne źródło prądowe

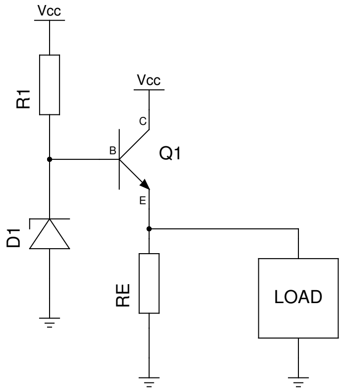
Przy pomocy wzmacniacza operacyjnego i tranzystora jesteśmy w stanie zrealizować precyzyjne (odporne na wahania temperaturowe itp) źródło prądowe, tak prądu wpływającego (NPN) jak i wypływającego (PNP). Ponadto możemy w łatwy, napięciowy sposób regulować prąd tego źródła (poprzez zmianę V_{ref}.



Dioda Zenera

Najprostsze stabilizatory napięcia można zrealizować w oparciu o diodę Zenera. Niestety spadek napięcia na diodzie jest funkcją prądu, ponadto najprostszy układ bez tranzystorowy charakteryzuje się znaczną rezystancją wyjściową. Problemem może być także to, iż aby uzyskać nominalny spadek napięcia trzeba puścić zazwyczaj przynajmniej 5mA, a jednocześnie należy uważać aby nie przekroczyć maksymalnej mocy która może wydzielić się na diodzie.





Obliczanie elementów (stabilizator bez tranzystora):

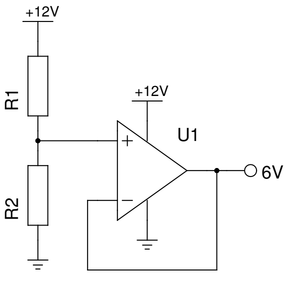
1. $R_1 \leq \frac{V_{CC_{min}} - U_Z}{I_{Z_{min}}}$
2. $R_{WY} = R_1 || r_d$
3. $S_u = \frac{\Delta U_{WY}}{\Delta V_{CC}} = \frac{r_d}{R_1 + r_d}$
4. wartość R_O wpływa na R_1 (w warunku na nie musimy uwzględnić odpływ prądu na obciążenie $I_{Z_{min}} \rightarrow I_{Z_{min}} + I_{O_{max}}$)

Obliczanie elementów (stabilizator z tranzystorem):

1. z założenia o $I_{E_{min}}$ otrzymujemy wartość $R_E = U_{WY}/I_{E_{min}}$
2. $R_1 = \frac{V_{CC_{min}} - U_Z}{I_{Z_{min}} + I_{O_{max}}/\beta}$
3. $R_{WY}(I) = r_d/\beta + r_{eb'}(I)$

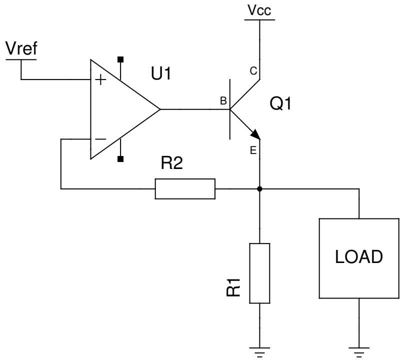
Źródło masy wirtualnej

Wspomniane wcześniej źródło masy wirtualnej można zrealizować np. na wzmacniaczu operacyjnym:



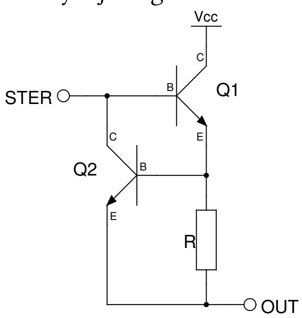
Stabilizatory z wzmacniaczem błędu

Na trochę zbliżonej do źródła masy wirtualnej zasadzie funkcjonuje stabilizator z wzmacniaczem błędu (będący podstawą popularnych stabilizatorów scalonych). Często w układach takich spotyka się zamiast pojedynczego tranzystora układ Darlingtona (z tego względu iż tranzystory mocy mają kiepskie bety). Ponadto zastosowanie R_2 umożliwia stosowanie źródeł referencyjnych o napięciu inne niż chcemy osiągnąć na wyjściu, co pozwala także na zmniejszenie ich prądożerności. Funkcję wzmacniacza błędu może pełnić np. wzmacniacz operacyjny lub różnicowy. Często spotkać można różnicowy z lustrem gdyż pozwala to na osiągnięcie małego drop-out'u (różnicy między minimalnym napięciem zasilania a napięciem wyjściowym).

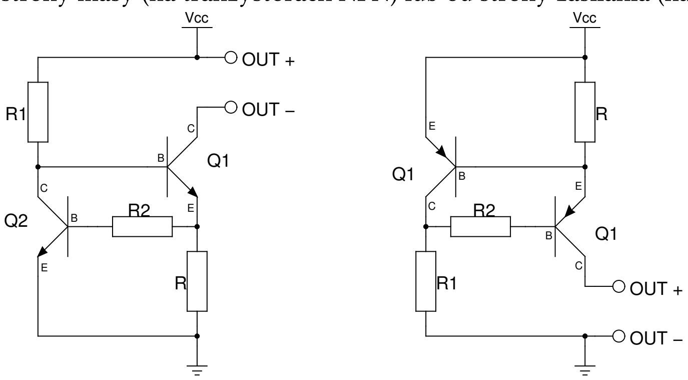


Ogranicznik nadprądowy

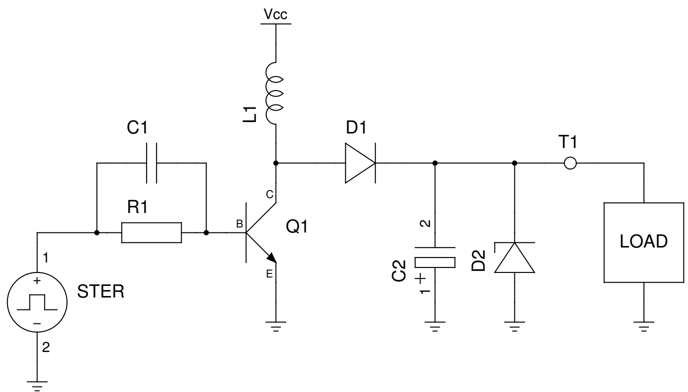
Istotnym aspektem układów zasilania jest ich zabezpieczenie przed poborem zbyt dużego prądu (funkcjonalność bezpiecznika elektronicznego). Układ taki można zrealizować np. w następujący sposób (nadmierny prąd powoduje nadmierny spadek na $R = \frac{0.6\mathrm{V}}{\mathrm{I_{max}}}$, co prowadzi do otwarcia Q2 i odciągania prądu z bazy Q1 - zakładamy że Q1 sterowany z jakiegoś źródła o niewielkiej wydajności prądowej - wzmacniacza błędu):



Poniższy schemat przedstawia typowe realizacje samodzielnych ograniczników prądowych[176] włączanych od strony masy (na tranzystorach NPN) lub od strony zasilania (na PNP):

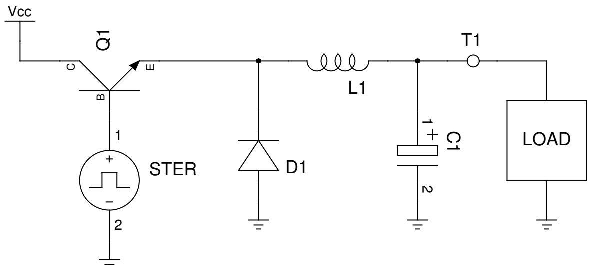


Przetwornica step-up



Działanie przedstawionej przetwornicy podwyższającej oparte jest na ładowaniu prądem cewki L1 (w czasie gdy Q1 w stanie nasycenia) oraz wyciąganiu z niej tego prądu celem ładowania C2 (w czasie gdy Q1 w stanie zatkania). W sytuacji gdyby nie było obciążenia oraz D2 napięcie na kondensatorze chciałoby wzrosnąć do nieskończoności, prowadząc do jego przebicia, lub przebicia Q1. Dlatego stosujemy D2 jako zabezpieczenie przed brakiem obciążenia. C1 przyspiesza przechodzenie Q1 w stan nasycenia/zatkania. $\frac{U_{WY}}{U_{WE}} = \frac{t_1}{t_2}$ $\quad I_{WY_{sr}} = \frac{\Delta I}{2} \cdot (1 - \frac{t_1}{t_1+t_2})$, gdzie ΔI - prąd do którego ładujemy cewkę, t_1 - czas ładowania, t_2 - czas rozładowywania. Ze względu na wielkość cewki chcielibyśmy jak największą częstotliwość pracy. Jednak jej ograniczeniem jest prędkość tranzystora, który ponadto musi wytrzymać U_{wy} i ΔI. Warto jednak zadbać aby układ pracował z częstotliwością nad akustyczną (inaczej będzie buczeć).

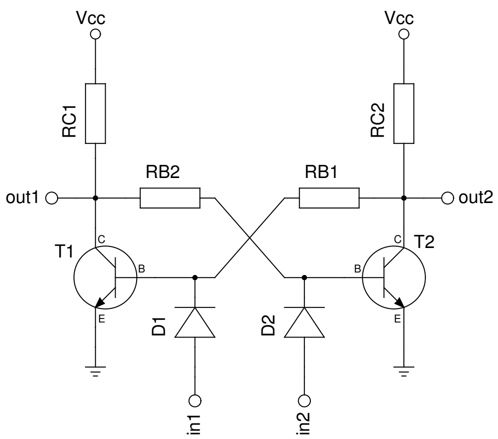
Przetwornica step-down



Działanie przedstawionej przetwornicy opiera się na takiej samej zasadzie jak ściemniacze elektroniczne, czyli na załączaniu i wyłączaniu obwodu na odpowiednie jednostki czasu, aby obniżyć wartość średniego napięcia wyjściowego do zadanej. Sterowanie takie nadaje się bez problemowo do urządzeń takich jak grzałki, żarówki itp. Zastosowanie takiego mechanizmu do zasilania elementów o małej bezwładności (układów elektronicznych) wymaga jednak wygładzenia skoków napięcia (nie możemy robić przerw w zasilaniu tylko musimy uśrednić wartość tego napięcia. Wymaga to dodania kondensatora filtrującego C1, jego dodanie powoduje niestety trudności z sterowaniem Q1 jako klucza nasyconego - stąd L1 i D1. Przetwornica tego typu przy prądzie poniżej założonego będzie miała wyższe napięcie na wyjściu od założonego (może osiągnąć nawet napięcie zasilania). $U_{WY} = U_{WE}\frac{t_1}{t_1+t_2}$

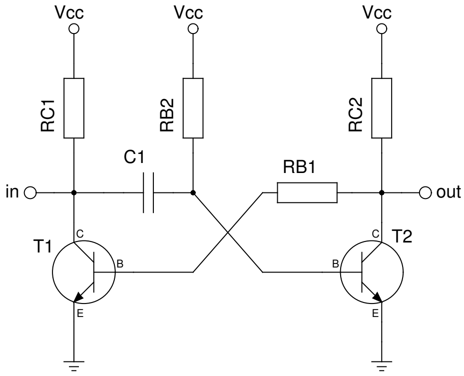
Przerzutniki

Bistabilny



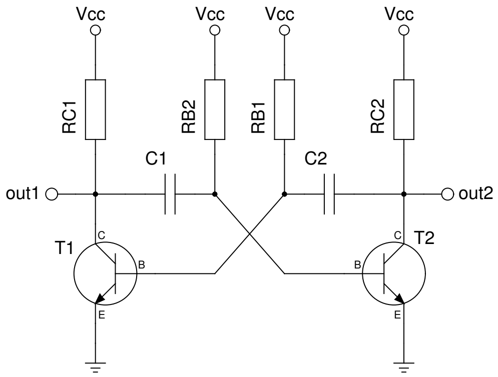
Układ ten służy do zapamiętania stanu binarnego. Może być przełączony poprzez podanie krótkiego sygnału na którejś z wejść lub zwarcie któregoś wyjścia do masy. powoduje to rozpoczęcie przewodzenia przez wybrany z tranzystorów, co prowadzi do spadku napięcia na bazie drugiego tranzystora, prowadząc do jego zatkania i trwałego ustalenia stanu wysokiego na bazie wybranego tranzystora.

Monostabilny



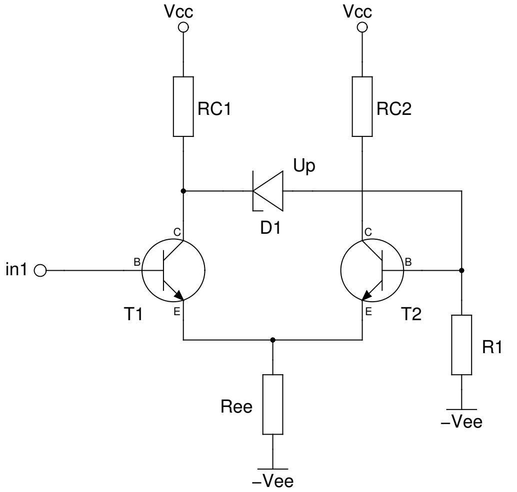
Układ służy do generacji impulsu trwającego ustaloną długość. Działa on podobnie jak przedstawiony przerzutnik bistabilny, z taką różnicą iż krótkotrwałe zwarcie wejścia do masy powoduje obniżenie potencjału bazy T2 o napięcie do jakiego został naładowany C1 (w normalnym stanie $U_{CC} - U_{BE}$. Powoduje to zatkanie T2 i przewodzenie T1, następuje także ładowanie C1 z stopniowym zwiększaniem napięcia na bazie T2 aż do osiągnięcia sytuacji pierwotnej. Należy zwrócić uwagę że po zatkaniu T1 (zaprzestaniu generowania sygnału wyjściowego) układ nie osiągnął jeszcze gotowości do generacji następnego impulsu gdyż musi nastąpił ładowanie C1 przez RC1 (należy odczekać $t \gtrsim 5\tau = 5R_{C1}C_1$). Wadą takiego rozwiązania są duże ujemne napięcia na bazach tranzystorów w chwili przerzutu, co (przy większych napięciach zasilania) może doprowadzić do ich przebicia.

Astabilny



Przerzutnik ten służy do generacji sygnału prostokątnego o zadanym wypełnieniu. poszczególne półokresy wynoszą: $t_x = \tau \ln(\frac{u(\infty)-u(0)}{u(\infty)-u(t_x)})$, gdzie $u(0)$ - napięcie początkowe asymptota procesu, $u(\infty)$ - asymptota procesu, $u(t_x)$ - napięcie w chwili zakończenia, $\tau = C_i R_{B_j}$ - stała czasowa elementów podłączonych do bazy tranzystora wyjście którego rozważamy w stanie wysokim. Działanie układu jest podobne do opisanego przerzutnika montostabilnego z tym że po (ponownym) nasyceniu drugiego tranzystora powstały spadek napięcia rozpoczyna taki sam proces dla pierwszego tranzystora i kondensatora podłączonego do jego bazy. Dodatkową wadą tego rozwiązania jest problem startu przy wypełnieniu 1/2 (jednakowych elementach) i łagodnym włączaniu.

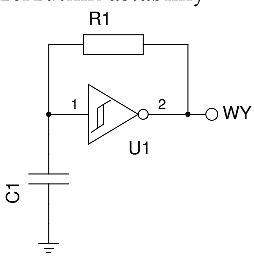
Shmitta



Służy od do uzyskania histerezy - większe napięcie powoduje przejście w stan wysoki, a mniejsze od niego w stan niski. Pozwala to m.in. na odszumianie sygnałów. Idea działania polega na zmianie potencjału drugiego wejścia wzmacniacza różnicowego zależnie od jego stanu. Przesuwnik poziomów zrealizowany na D1 oraz R1 służy do dostosowania napięcia wyjściowego do zakresu histerezy (wynosi on $U_{CC} - U_p$ - wartość górna, $U_{cc} - I_{ee}R_{C1} - U_p$ - wartość dolna). Występujący efekt millerowski na T2 w tym przerzutniku można usuwać np. poprzez zrealizowanie przesuwnika poziomów na wtórniku emiterowym i diodzie Zenera.

Zewnętrzna pętla opóźnienia

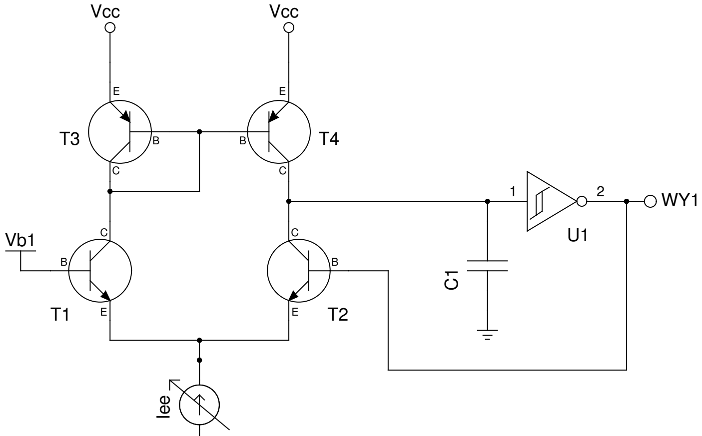
Podobnie jak przerzutnik Shmitta działają np. bramki cyfrowe tego typu. Bramkę taką możemy wykorzystać jako przerzutnik astabilny:

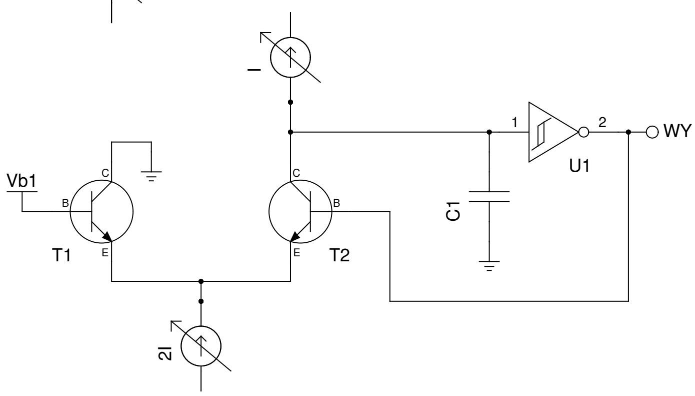


Korzystamy tutaj z histerezy tej bramki (napięcie na kondensatorze zmienia się pomiędzy tymi wartościami, a napięcie wyjściowe zgodnie z stanami logicznymi bramki. Długości półokresów wynoszą określa $t_x = \tau \ln(\frac{u(\infty)-u(0)}{u(\infty)-u(t_x)})$, gdzie $\tau = R_1 C_1$.

VCO - Generatory sterowane napięciem

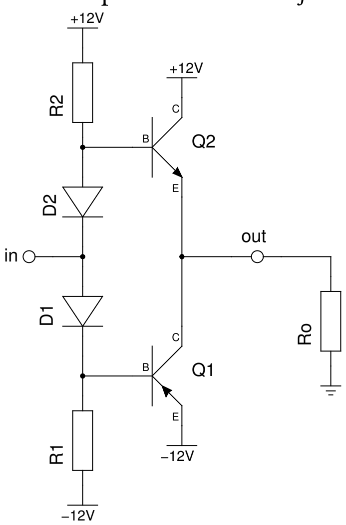
Idea tego typu układów polegna na napięciowym (na ogół prądowym, a dopiero prąd ustalany napięciowo) sterowaniu częstotliwością generowanego sygnału. Realizowane jest to poprzez wpychanie lub wyciąganie prądu z kondensatora C1 podłączonego do wejścia układu z histerezą. Wpychanie i wyciąganie prądu może być zrealizowane na rozmaite sposoby - np. tak jak przedstawiono na poniższym schemacie poprzez wzmacniacz różnicowy z lustrem lub wzmacniacz różnicowy z dwoma źródłami prądowymi (jedno będące wielokrotnością drugiego.

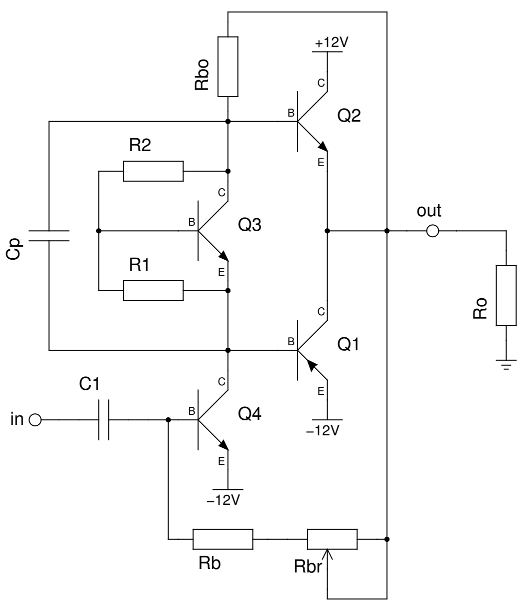




Wzmacniacz mocy

Trudnościami w realizacji liniowych wzmacniaczy mocy jest konieczność operowania na dużych wartościach napięcia i prądu, minimalizacji wydzielania mocy ($P = UI = I^2R = U^2/r$), przy jednoczesnym zachowaniu możliwie niezniekształconego sygnału oraz sporego wzmocnienia i rezystancji wejściowej. Podstawowym układem na którym można oprzeć tego typu rozwiązania jest wtórnik komplementarny (dwa wtórniki emiterowe połączone emiterami) - w odróżnieniu od zwykłego wtórnika emiterowego pozwala uniknąć problematycznego w tych zagadnieniach opornika emiterowego. Niestety układ ten cechuje się strefą martwą przy przełączaniu co powoduje konieczność zastosowania rozsównika poziomów. Na poniższym schemacie przedstawiono dwa warianty (prosty - bez dbania o drop-out oraz bardziej rozbudowany) takiego wzmacniacza.





Należy zapewnić diodom lub tranzystorowi robiącego za rozsównik napięć taką samą temperaturę jak tran-

zystorom mocy Q1 i Q2. Cp wspomaga rozsównik przy skokach. Rb i Rbr zapewniają poprawną pracę T3. Rbo robi za ”górne” źródło prądowe na zasadzie bootstrapu.

Najczęściej spotykanymi w dzisiejszych czasach wzmacniaczami mocy są wzmacniacze impulsowe, których działanie z grubsza polega na załączaniu w stan nasycenia tranzystorów odpowiedzialnych za dostarczenie kolejnej porcji prądu.

Technika cyfrowa

Współcześnie bardzo istotnym fragmentem elektroniki, odpowiedzialnym za bardzo wiele znanych z codziennego życia urządzeń (komputery, sterowniki, ...) jest elektronika cyfrowa[177]. Korzysta ona z skwantowanej do dwóch stanów postaci sygnału - stan wysoki i niski (w przeciwieństwie do analogowej, gdzie sygnał był ciągły - przyjmował dowolne wartości z obranego zakresu). Dlatego też tak ważne dla niej są Algebra Boola oraz kodowanie informacji w systemie dwójkowym.

System dwójkowy

System dwójkowy[178] jest (tak jak dziesiętny) systemem liczbowym, operuje on tylko na dwóch cyfrach - 0 i 1 (a nie jak dziesiętny na dziesięciu 0, 1, 2, ... 9) podstawą są potęgi liczby dwa zatem liczba w zapisie dwójkowym o postaci: dcba będzie odpowiadała liczbie dziesiętnej równej d * 2 + c * 2 + b * 2 + a * 2. Liczby systemu dwójkowego prosto zapisać jest w systemie 8 i 16 poprzez grupowanie w jedną cyfrę nowego systemu odpowiednio 3 i 4 cyfr z liczby zapisanej dwójkowo (licząc od prawej).

Algebra Boole'a i bramki logiczne

Algebra Boole'a[179] jest algebrą dwuwartościową (0 i 1) wyposażoną w trzy działania: sumy, iloczynu i negacji. W języku elektroniki tym podstawowym działaniom odpowiadają odpowiednio bramki: OR, AND, NOT. Bramki logiczne są układami elektronicznymi realizującymi różne funkcje logiczne. Działanie takich funkcji (jak również bramek elektronicznych) przedstawiane jest zwyczajowo w postaci tabelki prawdy.

Istnieje kilka sposobów zapisu liczb binarnych. Dla liczb dodatnich najpopularniejszym jest NKB[180], oprócz tego można spotkać się z kodem "1 z n"[181], kodem Graya[182], czy też kodem Johnsona[183]. Dla liczb ujemnych są to kod uzupełnień do dwóch[184] (U2), kod znak-moduł[185], czy też kod uzupełnień do jedności[186] (U1). Jeszcze innym zagadnieniem jest kodowanie liczb zmiennoprzecinkowych[187].

Podstawowe prawa algebry dwójkowej

$$a + 1 = 1 \qquad a \cdot 1 = a$$
$$a + 0 = a \qquad a \cdot 0 = 0$$
$$a + a = a \qquad a \cdot a = a$$
$$\bar{0} = 1 \qquad \bar{1} = 0$$
$$\bar{\bar{a}} = a$$
$$(a + b) + c = a + (b + c) \qquad (a \cdot b) \cdot c = a \cdot (b \cdot c)$$
$$a + b \cdot c = (a + b) \cdot (a + c) \qquad a \cdot (b + c) = a \cdot b + a \cdot c$$
$$a + b \cdot a = a \qquad a \cdot (b + a) = a$$
$$\overline{a + b} = \bar{a} \cdot \bar{b} \qquad \overline{a \cdot b} = \bar{a} + \bar{b}$$

konwersja AND/OR na NAND: $ab + cd = \overline{\overline{ab + cd}} = \overline{\overline{ab} \cdot \overline{cd}}$

konwersja OR/AND na NOR: $(a + b) \cdot (c + d) = \overline{\overline{(a + b) \cdot (c + d)}} = \overline{\overline{a + b} + \overline{c + d}}$

XOR: $a \oplus b = a\bar{b} + \bar{a}b$

$a \oplus 0 = a \qquad a \oplus a = 0$

Logika dodatnia a ujemna

W elektronice wyróżnić możemy dwie postaci logiki: dodatnią gdzie logicznej jedynce odpowiada stan wysoki (zazwyczaj realizowany przez brak napięcia) a logicznemu zeru stan niski (zazwyczaj brak napięcia) oraz logikę ujemną gdzie jest na odwrót, jak łatwo zaważyć tłomaczenia jednej logiki na drugą może dokonać bramka NOT.

Podstawowe bramki logiczne

Bramka logiczna[188] jest układem elektronicznym realizującym określoną funkcję logiczną. Bramki najczęściej realizowane są jako układy scalone zawierające w sobie kilka bramek.

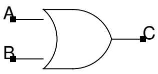
Tabelki prawdy dla funkcji OR[189], AND[190] oraz ich symbole elektroniczne:

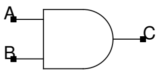
OR

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

AND

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

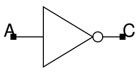




Tabelka prawdy dla funkcji NOT[191] oraz jej symbol elektroniczny:

NOT

A	C
0	1
1	0



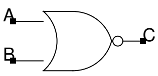
Tabelki prawdy dla funkcji NOR[192] (zaprzeczony OR), NAND[193] (zaprzeczony AND) oraz ich symbole elektroniczne:

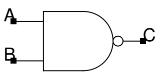
NOR

A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

NAND

A	B	C
0	0	1
0	1	1
1	0	1
1	1	0





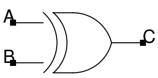
Tabelki prawdy dla funkcji XOR[194] (symetryczny OR), NXOR[195] (zaprzeczony XOR) oraz ich symbole elektroniczne:

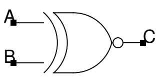
XOR

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

NXOR

A	B	C
0	0	1
0	1	0
1	0	0
1	1	1

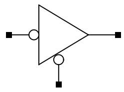




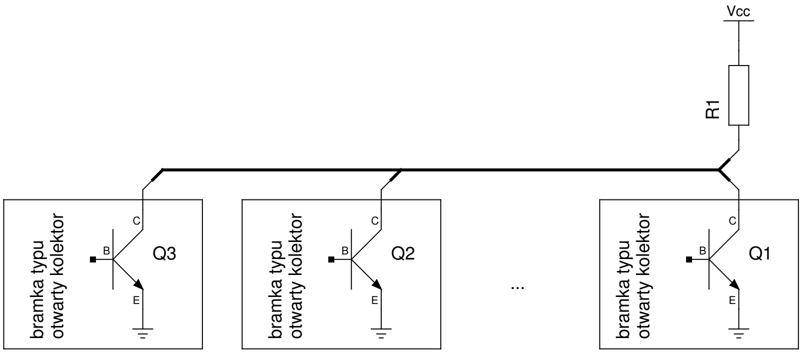
Dowolną funkcję logiczną można złożyć wyłącznie z bramek AND OR i NOT, a nawet wyłącznie z samych

bramek NAND albo NOR. Co ciekawe bramki te można zrealizować nie tylko w sposób elektroniczny ale także i mechaniczny (dowód - bramki logiczne z Lego[196]), a więc całą elektronikę cyfrową, łącznie z komputerami, mikrokontrolerami itd. możnaby wykonać na bazie kółek zębatych … . Należy zaznaczyć także występowanie bramek (i innych elementów cyfrowych) w wariantach:

— **trójstanowym**[197] - układ posiada dodatkowe wejście decydujące o tym czy wyjście jest podłączone czy też znajduje się w stanie **wysokiej impedancji** (wtedy układ nie wymusza na nim żadnego poziomu - nie wpływa na nie, a linią do którego jest podpięte może sterować inny układ.



— **otwarty kolektor**[198] (OC) - inny wariant realizacji układów mogących współsterować linią - w tym wypadku żadna z bramek podpiętych (właśnie kolektorem swojego stopnia wyjściowego) do linii nie podciąga jej do stanu wysokiego, ale każda może wymusić stan niski. Linia w tym wypadku wymaga zewnętrznego rezystora podciągającego.



Na koniec warto także wspomnieć o bramkach AOI (And-Or-Invert) będących od strony logicznej połączeniem dwóch bramek AND przy pomocy bramki NOR, tyle że zrealizowany w jednym układzie

Podstawowe przerzutniki

Poza opisanymi bramkami trzeba jeszcze koniecznie wspomnieć o przerzutnikach[199], które są podstawowymi elementami pamięci - najkrócej mówiąc przerzutnik ustawia odpowiedni stan wyjścia w w zależności od otrzymanego sygnału na wejściu i pamięta go do póki nie zostanie wymuszona jego zmiana. Przerzutniki posiadają dwa wyjścia: Q oraz wyjście -Q, na którym jest sygnał zaprzeczony w stosunku do Q.

Najprostszym rodzajem przerzutnika jest przerzutnik **RS** - po podaniu na wejście Set zera ustawia on wyjście Q na 1, natomiast po podaniu zera na wejście Reset ustawia on zero na wyjściu Q. Bardziej złożonym typem przerzutnika jest przerzutnik **D** - posiada on dodatkowo wejścia sygnałowe D oraz zegara C, w momencie narastającego zbocza sygnału C na wyjściu Q ustawiana jest wartość D. Kolejnym typem jest przerzutnik **JK** -zamiast wejścia D posiada wejścia J i K, zczytuje on stany wejść J i K w momencie nadejścia narastającego zbocza zegara (C) a wyjście ustawia przy zboczu opadającym, jeżeli J i K są w stanie zero to stan wyjścia nie zmieni się, jeżeli są w stanie 1 to wyjście zostanie zaprzeczone, jeżeli mają różne wartości to na wyjście zostanie przepisany stan J. (Zobacz w Wikipedii: przerzutnik typu RS[200], przerzutnik typu D[201], przerzutnik typu JK-MS[202])

Na schematach przerzutniki oznaczamy jako prostokąty z odpowiednio oznaczonymi wyprowadzeniami. Niekiedy potrzebujemy aby przeżutnik reagował na przeciwne zbocze - wyustarczy wtedy zanegować sygnał zegara. Poprzez łączenie kolejnych przerzutników jesteśmy w stanie uzyskać liczniki impulsów, rejestry, itp. Podobnymi do przerzutników elementami są **zatrzaski** (Latch) - reagują one na poziom sygnału a nie na zbocze, zatem występuje w nich faza przezroczysta.

Inne bramki logiczne i przerzutniki

W przypadku standardowych bramek niedopuszczalne jest łączenie ich wyjść - jest to możliwe po zastosowaniu bramek **z otwartym kolektorem** (oznaczanych przez dodanie kółka wewnątrz symbolu), wymagane jest wtedy podłączenie do wyjścia zasilania przez odpowiedni rezystor. Należy jeszcze wspomnieć o **bramkach Schmita**

- wyróżnia ona kierunek zmian sygnału (pętla histerezy - poziom sygnału powodujący zmianę stanu wyjścia zależy od tej zmiany) i służy głównie do wprowadzania sygnałów analogowych. Wyróżnia się także **bramki tórjstanowe** - załączają one wyjście pod wpływem dodatkowego impulsu sterującego.

Oprócz opisywanych przerzutników bistabilnych istnieją również przerzutniki **monostabilne** - podają one sygnał wejściowy na wyjście, jednak z opóźnieniem wynikłym ze stałej czasowej RC oraz **astabilne** - generują one sygnał prostokątny o częstości ustalanej przez stałą RC.

Zobacz także w Wikipedii: multiplekser[203], demultiplekser[204], pamięć cyfrowa[205], Programmable Logic Device[206], matryca CCD[207]

Przetworniki analogowo-cyfrowe i cyfrowo-analogowe

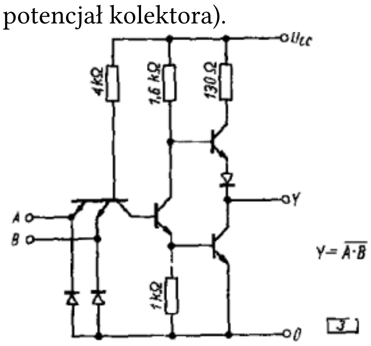
Poprzez odpowiedni układ sterowanych cyfrowo linii obciążonych różnymi rezystorami jesteśmy w stanie zamienić liczbę zapisaną binarnie na odpowiadające jej napięcie - jest to przetworzenie sygnału cyfrowego na analogowy (przetwornik DAC). Do przetwarzania sygnałów analogowych na cyfrowe służyć mogą **kompaktory analogowe** - gdy napięcie na wejściu + przekroczy napięcie na wejściu odniesienia (oznaczanym -) na wyjściu pojawia się 1, z ich wykorzystaniem (oraz opcjonalnie z wykorzystaniem przetwornika DAC) jesteśmy w stanie próbkować sygnał analogowy i zamieniać go na liczbę binarną.

Jak działa bramka?

Jak działa bramka TTL?

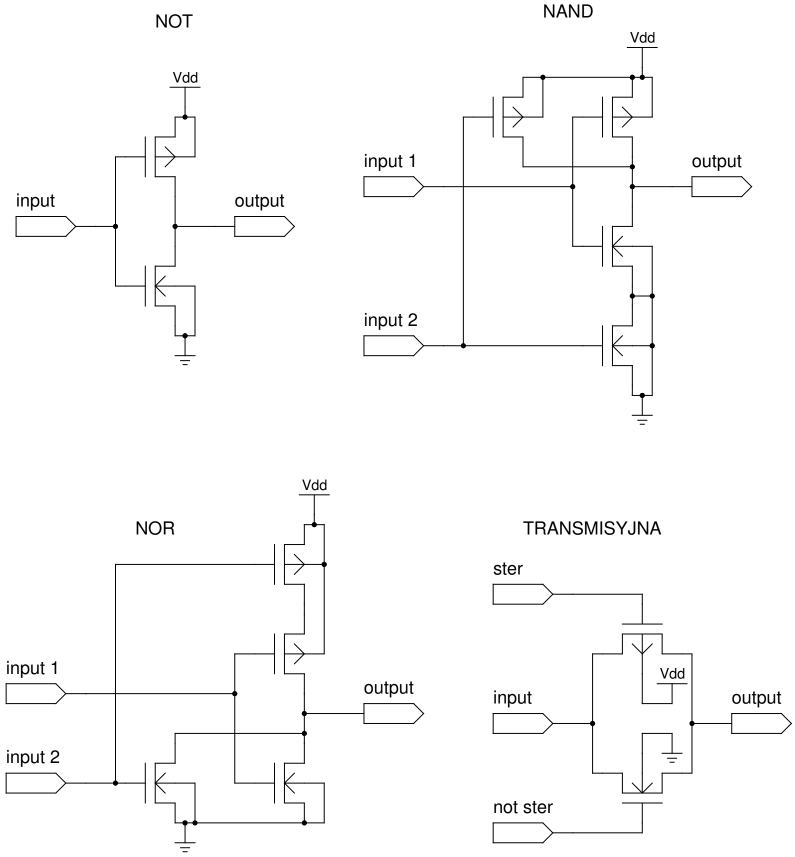
Poniżej przedstawiam schemat ideowy bramki NAND wykonanej w technologii TTL[208] (pochodzący z katalogu nieistniejącego już dzisiaj Naukowo-Produkcyjnego Centrum Półprzewodników CEMI z 1977 roku). Zadaniem tego układu jest podanie stanu niskiego (bliskiego 0) na wyjście Y gdy oba wejścia są w stanie wysokim (bliskim Ucc) oraz podawanie stanu wysokiego na Y w pozostałych przypadkach.

Gdy oba wejścia podłączone są do stanu wysokiego (lub są niepodłączone) w skutek podłączenia bramki dwuwemiterowego tranzystora NPN do stanu wysokiego jego kolektor ma potencjał wysoki (prąd odpływa z bramki przez kolektor), w skutek tego tranzystor na środkowej gałęzi zostaje wprowadzony w stan przewodzenia. To powoduje wprowadzenie w stan przewodzenia dolnego tranzystora na prawej gałęzi i podanie zera na wyjście. Jeżeli natomiast do jednego z emiterów tranzystora stopnia wejściowego przyłożymy zero, wymusi to stan niski na bramce tranzystora środkowej gałęzi i jego zablokowanie (prąd bramki odpływa przez emiter obniżając potencjał bramki a zatem i potencjał kolektora).



Jak działa bramka CMOS?

Współcześnie prawie wszystkie układy scalone wykonywane są w technologii CMOS[209]. Schematy ideowe inwertera, dwóch podstawowych bramek (NOR i NAND) oraz bramki transmisyjnej w tej technologii przedstawiam poniżej.



Działanie tych bramek (za wyjątkiem transmisyjnej) polega na otwieraniu tranzystorów podłączonych do napięcia które chcemy otrzymać na wyjściu, a zamykaniu prowadzących do napięcia przeciwnego. Dzięki zastosowaniu tranzystorów PMOS polaryzowanych Vdd oraz NMOS polaryzowanych GND obie gałęzie operują na tym samym sygnale wejściowym (nie jest wymagana jego negacja). Szeregowe łączenie tranzystorów zapewnia że należy otworzyć oba aby otworzyć daną drogę, a równoległe że otwarcie danej drogi powodowane jest otwarciem pojedynczego tranzystora. Dzięki zastosowaniu technologi MOS i podłączaniu wejść bramki tylko do bramek tranzystorów wejścia praktycznie nie pobierają prądu (istotnym wyjątkiem jest chwila zmiany sygnału).

Działanie bramki transmisyjnej polega na przepuszczaniu lub nie (w zależności od stanu wejścia sterującego) sygnału z wejścia na wyjście. Bramka taka nie regeneruje sygnału. Ponadto w uproszczonym (jednotranzystorowym) rozwiązaniu prowadzi ona do degradacji sygnału wartość w przybliżeniu równą napięciu progowemu tranzystora.

Układy programowalne

Obecnie bardzo istotną rolę w elektronice (cyfrowej) odgrywają układy programowalne. Można wśród nich spotkać zarówno bardzo proste układziki, jak i bardzo rozbudowane systemy (często oparte o architekturę ARM[210], MIPS[211] lub x86[212] i dysponujące znaczną mocą obliczeniową).

Zasadniczo należy wyróżnić dwa rodzaje układów programowalnych:

- układy o programowalnej strukturze (PLD[213]) - programowany jest układ bramek, przerzutników, itp. "umieszczanych" w kości; zamiast wykonywanego kodu (sekwencji instrukcji) program (napisany w Hardware Description Language[214] - typowo VHDL[215] lub Verilog[216]) opisuje **strukturę** układu logicznego (połączenia bramek, tablice prawdy, etc), która następnie jest programowana w fizycznej kości
- układy procesorowe[217] (przetwarzające ciąg instrukcji)

Wśród układów o programowalnej strukturze wyróżnić należy:

- SPLD[218] (PLE[219] - programowalna matryca bramek OR, PAL[220] i GAL[221] - programowalna matryca AND z dodatkowymi bramkami OR (często także obudowana rejestrami i multiplekserami na wyjściach), PLA[222] - programowalne matryce AND i OR)
- CPLD[223]
- FPGA[224] - programowalny element pamięciowy (możliwość zdefiniowania dowolnej - na ogół 4 wejściowej - funkcji w każdym elemencie logicznym, programowalne połączenia między elementami logicznymi i pinami, itd)

Wśród układów mikroprocesorowych wyróżnić należy:

- niewielkie układy (np. AVR[225] AtMega <=32, jakieś PIC[226]) nie pozwalające na zastosowanie systemu operacyjnego (tworzona aplikacja bezpośrednio operuje na zasobach sprzętowych układu - jest dla niego dedykowana, wykożystuje niekiedy biblioteki implementujące specjalistyczne funkcjonalności jak tuxgraphics TCP/IP stack[227], FreeMODBUS[228])
- większe układy (np. AtMega128, ARM7[229]) programowanie z wykorzystaniem prostych systemów czasu rzeczywistego, takiego jak: NutOS[230], FreeRTOS[231], uCLinux[232], uIP[233] / Contiki[234], PicOS[235] (PicOS jest dość ciekawym systemem służącym do tworzenia wielowątkowych aplikacji na małych systemach, wydaje się że możliwe a nawet wartościowa byłaby próba uruchomienia go na platformie AVR. Idea wielowątkowości w tym systemie polega na wykorzystywaniu przez watki współdzielonego stosu i zawieraniu przez nie wywołań "entry point" w których może nastąpić przełączenie wątku.), etherrape[236]
- sterowniki PLC[237] - programowanie tablicy przejść/wyjść deterministycznego automatu skończonego[238]; wykonywany jest cyklicznie program polegający na: 1. odczycie stanu wejść, 2. ustaleniu nowego stanu automatu 3. ustaleniu stanu wyjść 4. realizacji procesów komunikacyjnych. Użytkownik przy pomocy jednej z kilku standardowych metod programuje algorytm ustalania stanu następnego oraz powiązanie stanu automatu z stanem wyjść
- ARM9, MIPS, ... (systemy wbudowane z prawdziwym, aczkolwiek lekko przykrojonym, systemem operacyjnym - jakiś Linux, *BSD, ECos[239], OpenWrt[240], Linux/MIPS[241])
- systemy oparte na DSP[242] (też "prawie normalny system", ale niekoniecznie)
- PowerPC[243], x86[244] -> Single-board computer[245], PC/104[246], ... (systemy wbudowane typu PC, w zasadzie bez ograniczeń programowych)

Współcześnie w przypadku układów typu PLD (podobnie zresztą jak ma to miejsce na rynku mikrokontrolerów) ważniejsza jest struktura i możliwości danej serii układów czy konkretnego przedstawiciela niż podział na grupy takie jak FPGA, CPLD, ... Możliwa a często konieczna w jednym systemie jest współpraca kilku wyżej przedstawionych technologi. Szczególnie interesującym wydaje się pomysł zastosowania FPGA (bez pamięci stałej, czyli konfigurowanego niezależnie po każdym restarcie) jako rekonfigurowalnego układu wspierającego system mikroprocesorowy (mikrokontroler czy nawet coś klasy PC), który byłby każdorazowo reprogramowany pod bieżące potrzeby przez zarządzający nim system procesorowy.

Systemy wbudowane

Układy programowalne powszechnie stosowane są także w systemach wbudowanych[247] oraz dostępnych na rynku urządzeniach elektronicznych. Przy wyborze takich urządzeń (zwłaszcza w przypadku: telefonów IP, routerów/bramek IP, kamer IP, rejestratorów cyfrowych, odtwarzaczy, tunerów dvb, switchy zarządzanych, sterowników PLC/sterowników automatyki domowej, itp.) warto zwrócić uwagę na dostępność źródeł programu w oparciu o który działa dane urządzenie. Często jest to nawet istotniejsze od schematów elektronicznych urządzenia (gdyż stosowanie układów programowalnych dość radykalnie schematyzuje i uprasza budowę urządzeń), ale miło by było oczywiście jakby także hardware był "open source".

Na rynku dostępnych jest również bardzo wiele rozwiązań deweloperskich opartych na płytce z procesorem ARM (ew. MIPS) i działającym na niej systemem Linux (i nie tylko). Pozwalają one na dość łatwe i tanie budowanie różnego rodzaju systemów monitoringu i automatyki opartych na TCP/IP (np. poprzez dodanie peryferiów po I2C[248] lub USB[249]). Pewną niedogodnością takich rozwiązań w stosunku co do dedykowanych platform rozproszonego wejścia wyjścia, czy też profesjonalnych sterowników jest trudność w zapewnieniu estetyki podłączeń kabelkowych takiego systemu. Natomiast pod względem elastyczności, możliwości i dostosowania do potrzeb danego zagadnienia często przewyższają one wielokrotnie droższe rozwiązania profesjonalne.

W temacie dostępności tego typu rozwiązań swego rodzaju rewolucja rozpoczęła się wraz z projektem OpenWrt[250] i routera Linksys WRT54G w 2003 roku. Udostępnienie zgodnie z wymogami GNU GPL opartych na Linuxie źródeł tego routera pozwoliło na tworzenie alternatywnych wersji jego oprogramowania, o znacznie większych funkcjonalnościach niż oryginalne. W następstwie tego alternatywne wersje firmware pojawiały się dla coraz to większej gamy urządzeń. Na rynku także przybywało innych rozwiązań opartych na otwarte oprogramowanie. Niestety często nie udostępniających źródeł sterowników do wykorzystywanego sprzętu, lub też na tyle skutecz-

nie blokowanych że nie możliwe okazywało się uruchomienie na nich alternatywnego firmware. Niekiedy także z powodu zbyt wysokiej ceny danego sprzętu lub jego małej popularności nie doczekiwał się on otwartego systemu.

Kolejnym etapem tej rewolucji było wypuszczenie w 2012 roku Raspberry Pi[251], który zapoczątkował kolejny etap tej rewolucji. Po jego sukcesie masowo zaczęły pojawiać się inne platformy deweloperskie oparte o System-on-a-chip[252] (typowo z rdzeniem ARM) deklarujące wprost wsparcie dla Linuxa (a także innych systemów FLOSS[253]). Zdecydowanej poprawie uległa ich dostępność handlowa (dość szeroki wybór alternatywnych rozwiązań, brak konieczności szukania czy dany model da się przeflashować alternatywnym firmware, etc), cenowa (kompletny system komputerowy za mniej niż 100PLN) oraz użytkowej (dość przyzwoity poziom wsparcia, dokumentacji, etc).

Oprócz, wspomnainego już, znanego chyba wszystkim Raspberry Pi, wśród tego typu systemów deweloperskich należy wymienić:

— Banana Pi[254] (na szczególną uwagę zasługuje np. Banana Pi R1[255] z wbudowanym zarządzalnym switchem)
— BeagleBoard[256]
— Cubieboard[257]
— OLinuXino[258]
— HummingBoard[259]
— Orange Pi[260]
— MMnet1001[261] / MMnet1002[262]

Wartym uwagi są także układy konwersji USB na UART, inne protokoły lub GPIO, w szczególności FT4232[263] pozwalający uzyskać 4 interfejsy z których każdy może zostać skonfigurowany i użyty niezależnie jako: RS232, RS485/RS422, I2C, SPI, JTAG, GPIO. Niestety układy FTDI dostępne są tylko do montażu powierzchniowego, ale z pomocą przychodzą stosowne moduły (np. MMusb4232H[264]). Układ może być obsługiwany przez standardowe sterowniki w jądrze, własne sterowniki w jądrze oraz libusb[265].

Zobacz także: ewolucja koncepcji projektów inteligentnego domu, DVB STB: Enigma2, Rescue Avtech KPD674 after firmware upgrade fail, Open Hardware[266], Open Graphics Project[267], OpenSPARC[268], RepRap[269], OpenMoko[270], LEON[271].

Komputery przemysłowe

Nazwą tą określa się systemy wbudowane stosowane w różnego rodzaju sterownikach, kontrolerach itp. Rozwiązaniem spotykanym w zasadzie tylko w tego typu systemach jest architektura oparta o karty procesorowe lub wręcz komputery jedno-płytkowe (SBC[272]) i płyty główne zawierające wyłącznie magistralę - backplane[273]. Pewnym szczególnym rodzajem SBC są systemy "computer on module[274]". Warto tu wspomnieć iż w rozwiązaniach przemysłowych nadal często stosuje się standard magistrali ISA[275] wyparty prawie całkowicie z komputerów domowych ze względu na zbyt małą prędkość. Na koniec należy jeszcze wspomnieć iż rozwiązania "przemysłowe" przystosowane są często do zasilania niskonapięciowego, wykonywane są często także w różnych wariantach obudów - rack 19", 10", szyna DIN, montaż naścienny, itd.. Zobacz także: PC/104[276], CompactPCI[277], PXI[278], VMEbus[279], VXI[280], PICMG[281], KIM-1[282], ETX (form factor)[283], COM Express[284], XTX[285].

Mikrokontrolery

mikrokontrolery[286] są to programowalne, scalone układy cyfrowe integrujące w sobie jednostkę centralną (CPU), pamięć (zarówno operacyjną jak i służącą do przechowywania programu oraz danych) oraz układy wspomagające takie jak przetworniki analogowo-cyfrowe, … .

W dziale tym zajmiemy się głównie układami opartymi na mikrokontrolerach AVR[287] oraz programowaniem tych układów w C (z elementami asemblera). Na wstępie zachęcam do zapoznania się z instrukcją modułu zawierającego mikrokontroler ATMega128, dodatkowy RAM (z zewnętrznym kontrolerem pamięci) oraz kontroler ethernetowy: MMnet02[288] (zwłaszcza opis działania kontrolera pamięci i metody podłączenia do wyświetlacza LCD - uświadamia też pewne aspekty działania PC). Zachęcam też do zajrzenia do artykułów o podstawach programowania w C i C++.

Przy projektowaniu układów z mikrokontrolerami należy:

— przewidzieć i wmontować co najmniej 2 (lepiej 3) diody sygnalizacyjne - pokazujące poziom pracy, wykonywaną funkcję, etc
— przewidzieć i wmontować zestandaryzowane złącze ISP (Kanada lub jakiś własny konsekwentnie przestrzegany standard)
— piny ISP wykorzystywać wyłącznie do podłączenia diód sygnalizacyjnych, inne wykorzystanie tych pinów dopuszczalne jest tylko wtedy gdy jest to absolutnie niezbędne, może to być:

    - komunikacja SPI z układem obsługującym "chip select" (w stanie resetu ma być nieaktywny)
    - GPIO przez multiplekser sterowany sygnałem resetu uC (bardzo nie zalecane)
- w przypadku stosowania magistrali równoległej zaleca się multipleksowanie linii adresowej i danych
- pamiętać że wiele uC potrafi podciągać linię wejście w górę ale nie w dół, więc przyciski powinny być zwierane do masy
- pamiętać że sterowanie lampkami sygnalizacyjnymi (zwłaszcza gdy wymagany jest do tego zewnętrzny tranzystor) prostsze jest od strony masy (tranzystor NPN), zatem stale podłączonym pinem takich kontrolek należy robić VCC (a nie masę); niestety w połączeniu z powyższym oznacza to iż do modułu z przyciskami i kontrolką oprócz linii sygnałowych musimy doprowadzić zarówno masę jak i biegun dodatni zasilania
- jeżeli korzystamy z modułów podłączanych jakąś magistralą (nieważne czy szeregową czy równoległą) umożliwić wymuszenie resetu takich modułów z poziomu mikrokontrolera (istotne także/zwłaszcza gdy mówimy o jakimś ARMie z systemem operacyjnym i np. Atmegach jako moduły)

Warto także bazować w swoich projektach na kilku mikrokontrolerach z jednej rodziny - w przypadku nie seryjnej produkcji oszczędności z zastosowania bardzo małego mikrokontrolera zostaną zatarte "kosztami" związanymi z trudnościami w jego użyciu (brak pinów do debugowania, mała pamięć programu, ...) oraz związanymi z rozwijaniem projektów na różnych mikroprocesorach (bo do kolejnego był już za mały). Osobiście polecam za minimum przyjąć Atmega8 (a jeżeli jest nie wystarczający to Atmega16). Niestety niektóre z starszych, porzuconych projektów przedstawionych na stronie są niezgodne z przedstawionymi wytycznymi dotyczącymi projektowania układów bazujących na mikrokontrolerach.

Potrzebne oprogramowanie i sprzęt:

- mikrokontroler z podpiętym czymś do wejść oraz wyjść
- kompilator avr-gcc (dostępny dla Linuxa oraz w WinAVR dla Windows) wraz z blibliotek avrc
- programator isp (np. USBasp[289], Programator USB z obsługą HVSP[290], Programator ATxmega za złotówkę, czyli obsługa PDI w USBASP[291]) lub jtag, niekiedy przydatny może okazać się także programator równoległy[292]
- debugger, symulator, ...

Kompilacja (programów w C dla AVR):

Kompilujemy i linkujemy: `avr-gcc -mmcu=atmega128 -Os -ggdb plik_zrodlowy.c -o plik_wynikowy.o`

Tworzymy plik hex:`avr-objcopy -O ihex plik_wejsciowy.elfplik_wynikowy.hex`

Debugowanie, ...:

O debugowaniu, dezasemblerowaniu i symulacji z wykorzystaniem narzędzi linuxowych napisałem w "Debugowanie (gdb)". Jednak opisany tam program do symulacji nie jest jeszcze idealny i może się zdarzyć że nie będzie działał poprawnie, drobne problemy miałem też z jtag. Przy szukaniu błędów w programie uruchamianym już w układzie bardzo przydatny jest port szeregowy przez który program może podawać co aktualnie robi, a my możemy wpływać na jego działanie. Przykłady wykorzystania portu szeregowego można zobaczyć m.in. w moim projekcie "Sterownik ogrzewania akwariowego". Poniżej przykład komend potrzebnych/przydatnych do debugowania przez jtag:

```
avarice --debug --part atmega128 --jtag /dev/ttyS0 :4242
avr-gdb simple.elf
        # target remote localhost:4242
        # break *0xaddr
        # backtrace
avr-objdump -d simple.elf
avarice --erase --program --verify --file simple.elf --jtag /dev/ttyS0
```

Nut/OS

Jednym z projektów dotyczących programowania mikrokontrolerów jest dystrybuowany na zasadach licencji typu BSD system czasu rzeczywistego z stosem TCP/IP - Nut/OS[293]. Poniżej zamieszczę krótki poradnik dotyczący wykorzystania tego systemu w wspomnianym wyżej module sieciowym Mmnet02.

1. instalujemy na komputerze Nut/OS - opis instalacji Nut/OS dla Debian[294]
2. budujemy biblioteki oraz drzewo przykładowych programów (np. przy pomocy nutconf - do NutOSa załączone są pliki opisujące konfigurację dla Mmnet02, w przypadku problemów z ściezkami warto zajrzeć do opcji programu)
3. w katalogu w którym utworzyliśmy drzewo z przykładowymi programami tworzymy podkatalog na nasz projekt i kopiujemy do niego plik Makefile np. z basemon i dostosowywujemy do swoich potrzeb (nazwę projektu, listę plików źródłowych, bibliotek itp.)
4. w katalogu naszego projektu tworzymy program ... zachęcam do skorzystania na początek z stworzonego przeze mnie nut_first.c[295] oraz do zapoznania się z przykładami w wygenerowanym drzewie (m.in. serwer http, ftp, ...).

NUT_FIRST.C

```
/*
 * plik ma na celu prezentację głównych możliwości systemu Nut/OS
 * oraz ułatwienie rozpoczęcia pracy z tym pakietem
 *
 */


#include <stdio.h>
#include <io.h>
#include <sys/timer.h>

// UART and Ethernet and LCD
#include <dev/board.h>
#include <dev/nicrtl.h>
#include <arpa/inet.h>
#include <sys/confnet.h>
#include <dev/hd44780_bus.h>

// bezposredni dostep do pamieci po adresie, w oparciu o funkcje nic_inlb, nic_outlb z
ethernut
#define mem_read_byte(reg) (*(volatile u_char *) (reg))
#define mem_write_byte(reg, val) (*(volatile u_char *) (reg) = (val))


// funkcja pobiera linię z UART
// jezeli podano out != NULL linia bedzie wypisana do tego pliku (np. na uart ...)
void UartGetLine(u_char *buff, int len, FILE *out) {
 int i=0;
 while (i<len) {
  while ((inb(USR) & 1<<RXC) == 0); // czekamy na znak
  buff[i] = inb(UDR);
  if (out!= NULL) fputc(buff[i], out);
  if (buff[i] == '\n')
   break;
  i++;
 }
 buff[i] = '\0';
}

// funkcja pobierajaca konfiguracje IP z UART
u_long UartGetConfig(u_char * msg, u_long val, FILE * out) {
 u_long tmp;
 char inbuff[32];
```

```
  while(1) {
   fprintf(out, "Podaj %s [%s]: ", msg, inet_ntoa(val));
   UartGetLine(inbuff, 32, out);
   if (inbuff[0]=='\0' || inbuff[1]=='\0') {
    fprintf(out, " ... bez zmian\n");
    return val;
   }
   tmp=inet_addr(inbuff);
   if (tmp != (u_long) -1) {
    fprintf(out, " ... zmieniono\n");
    return tmp;
   }
  }
}




void main(void) {
  //extern void *__bss_end;
  //NutHeapAdd(&__bss_end, (uptr_t) RAMEND - 256 - (uptr_t) (&__bss_end));
  //NutHeapAdd((void *) (RAMEND + 1), sram);
  /// konfiguracja trybu pracy kontrolera pamięci
  mem_write_byte(0xff01, 0x00);
  /// wylaczenie przetwornikow A/D
  sbi(ACSR, ACD);


/// uruchamianie RS-232
  NutRegisterDevice(&DEV_UART, 0, 0);
  outb(UCR, 1<<RXEN | 1<<TXEN); //| 1 << RXCIE;
   // włączenie nadajnika, odbionika UART wraz z generowaniem przerwań
  outb(UBRR, 51); // szybkosc transmisji 9600 bps
  FILE *uart = fopen(DEV_UART_NAME, "w");

/// wypisanie czegos przez RS-232
  fprintf(uart, "Hello World !!!\n");
  fprintf(uart, "connection detect ...\n");

/// wykrywanie polaczenia przez RS-232
  uint32_t i;
  char uart_is_use=0;
  for (i=0; i<5000000; i++) { // ok 10 sek.
   if ((inb(USR) & 1<<RXC) != 0) {
    uart_is_use=1;
    break;
   }
  }

 if (!uart_is_use) {
   fprintf(uart, "UART is NOT use !\n");
  } else {
   fprintf(uart, "UART is use !\n");

   /// oproznienie bufora
   while ((inb(USR) & 1<<RXC) != 0) inb(UDR);

   fprintf(uart, "Napisz cos:\n");
```

```
  /// wczytanie czegos przez RS-232
  char inbuff[32];
  UartGetLine(inbuff, 32, uart);
  fprintf(uart, "Napisales: %s\n", inbuff);
 }


/// konfigurator sieci
 fprintf(uart, "Konfigurowanie sieci\n");
 NutNetLoadConfig("eth0");
 u_long ip_add=confnet.cdn_cip_addr;
 u_long ip_mask=confnet.cdn_cip_addr;
 if (ip_add==0) {
  fprintf(uart, "  uzycie konfiguracji domyslnej\n");
  ip_add=0x6400A8C0; // 192.168.0.100
  ip_mask=0x0000ffff; // 255.255.0.0
 }

if (uart_is_use) {
  ip_add = confnet.cdn_cip_addr = UartGetConfig("IP  ", ip_add, uart);
  ip_mask = confnet.cdn_ip_mask = UartGetConfig("Mask", ip_mask, uart);
  fprintf(uart, "zapisac [y/*]\n");
  char inbuff[3];
  UartGetLine(inbuff, 3, uart);
  if (inbuff[0]=='y') {
   fprintf(uart, " zapisano\n");
   NutNetSaveConfig();
  }
 }


/// siec
 fprintf(uart, "Wczytana konfiguracja sieci:\n");
 fprintf(uart, "IP:     %s (%lx)\n", inet_ntoa(ip_add), ip_add);
 fprintf(uart, "Mask:   %s (%lx)\n", inet_ntoa(ip_mask), ip_mask);
 //jest tez: confnet.cdn_gateway oraz confnet.cdn_mac[0] ... confnet.cdn_mac[5]

fprintf(uart, "Uruchamianie sieci\n");
 if (NutRegisterDevice(&DEV_ETHER, 0x8300, 5))
  fputs("Registering device failed", uart);

if(NutNetIfConfig("eth0", NULL, ip_add, ip_mask))
 fprintf(uart, "Siec skonfigurowana\n");


/// lcd
 fprintf(uart, "LCD Start\n");
 NutRegisterDevice(&devLcdBus, 0xff04, 0);
 FILE *lcd = fopen("lcdbus", "w");
 fprintf(lcd, "Helo World !!!\n");

NutSleep(10000); // krotka przerwa

// na podstawie funkcji LcdWriteCmd(); z arch/avr/dev/hd44780_bus.c
 // w pliku tym sa takze LcdInit(); LcdWriteData(); ...
 // sterowac mozna poprzez sekwencje ESC albo bezposrednio komendami - jak nizej
 mem_write_byte(0xff04, 1 << LCD_HOME);

fprintf(lcd, "Witaj\n");
```

```
if(uart_is_use) {
  char inbuff[32];
  while(1) {
   UartGetLine(inbuff, 32, uart);
   fputs(inbuff, lcd);
   fflush(lcd);
  }
 }

while(1)
  NutSleep(10000);
}
```

ARM

ARM czyli Advanced RISC Machine jest zaawansowaną 32-bitową architekturą procesorów (mikrokontrolerów). Procesory tego typu bardzo często wykorzystywane są w systemach wbudowanych, routerach sprzętowych, telefonach IP itp.

Dużą zaletą w programowaniu większych układów ARM (a także innych tego typu mikrokontrolerów, procesorów, układów system-on-chip, np. MIPS[296]) jest to iż często posiadają one normalny system - dedykowana/okrojona (np. OpenWRT[297]), albo nawet typowa (np. w Raspberry Pi[298]) dystrybucja systemu Linux. Umożliwia to bardzo wygodne debugowanie kodu sterującego tworzonego na taką platformę, a także tworzenie go z wykorzystaniem standardowych posixowych rozwiązań (system plików, sieć, ...).

DSP - PRZETWARZANIE SYGNAŁÓW

Procesor DSP[299] są procesorami zoptymalizowanymi do przetwarzania ucyfrowionych sygnałów. Charakteryzują się możliwością równoczesnego wykonywania kilku operacji w ramach jednej instrukcji, sprzętowymi układami mnożącymi lub mnożąco-akumulującymi, rozdziałem (możliwością równoległych dostępów) pamięci próbek i współczynników. Do najistotniejszych cech charakteryzujących procesor sygnałowy od strony programistycznej (znaczną część tych cech można także rozpatrywać porównując architektury procesorów ogólnego przeznaczenia) należą:

- bitowość i typ procesora - zmiennoprzecinkowy, stałoprzecinkowy (ułamkowy lub zwykły), mieszanych
- długość bitowa instrukcji i wynikłe z niej ograniczenia (np. wielkość danych natychmiastowych, nieortogonalności w liście instrukcji, etc)
- bogactwo listy rozkazowej, rozkazy predykatowe (wykonywane warunkowo)
- ilość różnego typu rejestrów oraz poziom ich uniwersalności (które mogą być używane na wyjściu, które na wejściu, które do adresowania, etc - rejestry specjalizowane vs rejestry uniwersalne/ogólnego przeznaczenia)
- dostępne tryby adresowania rejestrowego (przy pomocy dedykowanych rejestrów adresowych, mogących być modyfikowanymi - inkrementowanymi, zwiększanymi o stałą wartość lub wartość z podanego rejestru, etc - bez dodatkowych narzutów), zauważalny rozrost jednostek generacji adresów do (prawie) normalnych ALU[300]
- rozdzielność / ilość przestrzeni adresowych (np. trzy - programu (P:), danych ”x” (X:) i danych ”y” (Y:))
- zorientowaność rejestrowa (konieczność ładowania danych z pamięci do rejestrów przed ich wykorzystaniem) lub jej brak
- przesłania pamięć-pamięć
- rozdzielony lub zintegrowany układ mnożący i ALU
- pętle sprzętowe vs. pętle programowe gdzie warunkowy skok zajmuje tylko ułamek częsci jednego słowa rozkazowego
- konstrukcja potoku wewnętrznego, itp
- system przerwań (wspieranie szybkiej, krótkiej obsługi przerwań, wielkość wektora przerwań, etc)
- mechanizm DMA
- ...

W wspomnianym przetwarzaniu sygnałów coraz istotniejszą rolę zaczynają odgrywać architektury wielordzeniowe, gdzie grupy rdzeni funkcjonują w trybie SIMD[301] lub SPMD[302]. Do głównych przedstawicieli tego nurtu

należą procesory CELL[303] i procesory graficzne GPU[304]. Tendencja ta wpływa także na wzrost znaczenia algorytmów umożliwiających prowadzenie obliczeń w sposób równoległy (np. algorytmów kratowych).

Linki i moje projekty

Zachęcam także do zapoznania się z moimi projektami związanymi z programowaniem mikrokontrolerów: powiadamiacz o IM dla jabberd2 (AVR NutOS), sterownik ogrzewania (AVR), sterownik oświetlenia (AVR), centralka alarmowa, konwerter RS do 8bit/OneWire/I2C/SPI.

Zobacz w Sieci: programowanie mikrokontrolera ATmega32[305] (m.in. opis liczników, komparatora, ...), radzio.dxp.pl[306], mikrokontrolery.net[307], Zbigniew Czaja - strona domowa[308], Craft[309] (gra wideo na ATmega88), Zablokowany ATMEGA8[310].

Zwiększamy ilość wejść/wyjść czyli magistrale

Typowe mikrokontolery posiadają pewien zasób GPIO[311] i innych interfejsów wbudowanych w siebie, w niektórych sytuacjach okazuje się on jednak niewystarczający - można wtedy podłączyć poprzez jakąś magistralę układy zwiększające ilość wejść/wyjść lub zastosować kilka mikrokontrolerów (często również komunikujących się poprzez jakąś magistralę). Zasadniczo mamy do wyboru:

- magistrala równoległa
  - składa się z: szyny danych, szyny adresowej (może to być jedna szyna multipleksowana sygnałem informującym o tym czy nadajemy dane czy adres) oraz sygnały sterujące transmisją - strobe i zapis/odczyt lub strobe zapisu i strobe odczytu
  - korzystamy z rejestrów i buforów adresowalne (wejścia adresowe może posiadać sam ten układ lub można stosować układy z pojedynczym wejściem enable i osobny dekoder adresu)
  - trochę na temat ich stosowania napisałem w artykule opisującym ewolucję koncepcji projektów inteligentnego domu
- rejestry przesuwne
  - jako input (74HC165)
    - zapamiętujemy stan wejść równoległych układu (sygnał PL = asynchronous parallel load input)
    - podajemy sygnał clock (CP = clock input) i w jego takt czytamy koleje bity z wyjścia szeregowego (Q7 = serial output from the last stage)
  - jako output (74HC595)
    - wystawiamy daną na wejście szeregowe (DS = serial data input)
    - dajemy clock przesunięcia rejestru (SHCP = shift register clock input)
    - po ustawieniu wszystkich bitów dajemy dajemy sygnał ustawienia wyjść na zadaną wartość (STCP = storage register clock input)
- magistrale szeregowe typu I2C, SPI, 1-wire, UART

Najprostsza w użyciu wydaje się magistrala równoległa bez multipleksowania szyny adresowej i danych (ustawienie adresu, zapis danych sterowane dwoma sygnałami), jednak wymaga ona stosowania dużej ilości połączeń między modułami (typowo minimalistycznie: 8 bitów szyny danych, 3-8 bitów szyny adresowej, 2 sygnały sterujące). W przypadku podziału projektu na moduły nastręcza to problemy z ich łączeniem - można/trzeba kombinować z tworzeniem jakiegoś backplane i stosowania jakiś złączy (np. złączy krawędziowych na PCB), ale jest to dość kłopotliwe w realizacji i przy małych modułach (a tym bardziej niewielkiej ich liczbie) trochę dziwne.

Przy konieczności dodania niewielkiej ilości modułów wejść wyjść mogą się sprawdzić rejestry przesuwne, ale w odróżnieniu od magistral szeregowych rozwiązanie to nie skaluje się dobrze.

Najbardziej elastycznym (ale i dość drogim) rozwiązaniem jest zastosowanie magistral szeregowych (zasadniczo będących bardziej rozwiniętą formą rejestrów przesuwnych). Na przykład wspomniana magistrala I2C umożliwia podpięcie na pojedynczej linii kilkunastu układów, umożliwia multipleksowanie swojej linii celem zwiększenia pojemności i zapewnia 8 bitowe adresowanie układów na magistrali (realizowane w protokole transmisji - bez dodatkowych połączeń w tym celu).

Podobnym zagadnieniem jest podłączanie klawiatur, ale tutaj możemy potraktować ją jako matrycę przełączników - i zasilać tylko kolejno pojedyncze rzędy szukając kolumny która jest zwarta (patrz manipulator alarmowy),

w przypadku gdy dopuszczamy równoczesne wciśnięcie kilku z tych przełączników należy stosować diody uniemożliwiające przeniesienie potencjału zasilonego wiersza poprzez linie kolumny do innych rzędów.

Interfejsy elektroniki

Praktycznie każdy z systemów elektronicznych korzysta z jakiś interfejsów komunikacyjnych. Podstawowym interfejsem jest magistrala (szyna danych i adresów), jej szczegółowe parametry zależą od konkretnego systemu. Szyna systemowa może składać się z osobnej szyny danych (linie oznaczane zazwyczaj D0-Dxx, gdzie xx to bitowość szyny danych) i adresowej (linie oznaczane zazwyczaj A0-Ayy), bądź wspólnych linii (AD) multipleksowanych osobnym sygnałem (AEN) informującym czy aktualnie nadawany jest adres czy dane. Szyna taka zawiera także linie informujące o przeprowadzanej operacji (WR i RD) ich funkcja może być także połączona w jednej linii. Któreś z sygnałów (AEN, WR, RD, lub jakieś inne) pełnią rolę tzw. strobu (zmiana ich stanu decyduje o rozpoczęciu ważności danych / adresów na liniach). W wielu mikroprocesorach szyna dostępna jest w dwóch trybach - pamięci i urządzeń I/O (o trybie pracy decyduje osobna linia). Niekiedy można się spotkać z memory maped I/O czyli udostępnianiem urządzeń w przestrzeni adresowej pamięci (nawet pomimo że architektura wspiera rozdzielenie przestrzeni I/O i pamięci). Warto wspomnieć także o technice DMA[312] umożliwiającej zlecenie wykonania operacji na szynie (np. kopiowania z pamięci do urządzenia) zewnętrznemu kontrolerowi (tak aby nie blokowały one procesora). Zauważalne jest obecnie odchodzenie od koncepcji magistrali (szyny systemowej) na rzecz łączy szeregowych punkt-punkt lub rozwiązań sieciowych opartych na przekazywaniu pakietów.

Dostęp do pamięci czy urządzeń podłączonych do szyny odbywa się za pomocą ich adresów - logika urządzenia musi kontrolować czy w danym cyklu szyny był wywołany jej adres (odpowiednia kombinacja bitów na szynie adresowej) czy nie. W przypadku podłączania standardowych układów często logikę tą konstruujemy samodzielnie i wykorzystujemy wejście chip select /chip enable (CS/CE) do aktywacji lub dezaktywacji układu. Urządzenia nadawcze powinny być wyposażone w bufory trój-stanowe, które dopóki dane urządzenie nie zostanie wywołane do odczytu odłączają je od szyny danych. Urządzenia odbiorcze powinny posiadać rejestry zapamiętujące stan linii danych (interesujących nas bitów z tej linii) w chwili strobu zapisu. Istotne jest także uwzględnianie czasu potrzebnego na skonsumowanie / bądź wystawienie na szynę odpowiednich danych (czas ten musi upłynąć zanim do układu peryferyjnego/pamięci będziemy mogli zapisać kolejną wartość lub zanim dane wystawiane przez układ na liniach danych będą ważne). Niektóre z urządzeń potrafią sygnalizować zakończenie bardziej czasochłonnych operacji (np. pamięci EEPROM[313] dopóki nie zakończą zapisu przy odczycie wystawiają zaprzeczenie zapisywanych danych na D7 lub zmieniają przy każdym kolejnym odczycie wartość D6).

Również często są wykorzystywane standaryzowane interfejsy (głównie szeregowe) takie jak:

- UART[314] (RS232[315], RS485[316], RS422[317], a także światłowodowo np. z wykorzystaniem modułów Toslink[318]) - uniwersalny asynchroniczny odbiornik i nadajnik, bardzo popularny format komunikacji szeregowej używającej dwóch linii komunikacyjnych (nadawanie i odbiór) oraz niekiedy dodatkowych linii sterujących transmisją, posiada kilka wariantów rozwiązań elektrycznych transmisji (ale konwersja pomiędzy nimi ma charakter czysto elektryczny, w mikrokontrolerach najpopularniejszy jest TTL 5V, rzadziej 3.3V), zarówno odbiornik jak i odbiornik muszą mieć ustawione takie same parametry transmisji (szybkość, znaczenie 9 bitu*, itp), interfejs występujący praktycznie we wszystkich mikrokontrolerach, komputerach jednoukładowych itp.

  *) Standardowym jego wykorzystaniem jest kontrola parzystości, ale jednym z ciekawszych zastosowań 9 bitu jest ustawianie go w bajcie adresowym, i konfigurowanie tylko nadajnika który odebrał swój adres do odbioru danych bez ustawionego 9 bitu (pozostałe nie będą wtedy odbierać danych przeznaczonych dla tamtego nadajnika, ale odbiorą kolejny adres nadawany z ustawionym bitem dziewiątym ...).
- interfejsy typowo służących połączeniu mikrokontrolera z czujnikami, modułami I/O, itd:
  - Serial Peripheral Interface[319] - interfejs często spotykany w mikrokontrolerach służący do komunikacji z innymi układami, oparty na zasadzie master-slave, wykorzystujący 3 linie (nadawanie, odbiór, zegar) magistralowe oraz zależną od zastosowania liczbę linii wyboru układu (Chip Select), standard wykorzystywany także w programowaniu ISP[320] dla AVR[321]
  - I2C[322] - magistrala dwuliniowa (linia danych i linia zegarowa), obsługująca 7 bitową adresację, umożliwiająca budowę bardziej rozległych systemów niż SPI, często występująca w mikrokontrolerach
  - 1-Wire[323] - jedno przewodowa magistrala (umożliwiająca także zasilanie układów z przewodu magistralowego), obsługująca 64-bitową adresację (adresy ustalane przez producenta układów peryferyjnych, powinny być unikalne w skali globalnej), często stosowana do komunikacji z czujnikami i pamięciami (w tym i-button), niestety praktycznie nigdy nie występująca "na pokładzie" mikrokontrolerów, zatem wymagająca implementacji softwarowej (z ostrymi rygorami czasowymi) lub stosowania układu mostka pomiędzy np. I2C a 1-wire
  - CAN[324] - popularna dwu przewodowa magistrala "przemysłowa" działająca na zasadzie rozgłoszeniowej, ob-

służąca 11-to lub 29-bitowe identyfikatory, doczekała się niemałej liczby protokołów na niej bazujących takich jak: CANopen[325], CAN Kingdom[326], MilCAN[327], SafetyBUS p[328]

- interfejsy typowo służących połączeniu komputera z urządzeniami peryferyjnymi:
  - USB[329] - popularny zwłaszcza w komputerach (a co za tym idzie urządzeniach które chcemy do nich podłączać) magistrala szeregowa, możliwa ale dość trudna w implementacji softwarowej częściej uzyskiwana za pomocą konwersji z UART, niekiedy wbudowana obsługa w mikrokontroler.
  - IEEE1394[330] - stosunkowo mało popularny interfejs, zwany też FireWire, występujący głównie w sprzęcie wideo oraz w niektórych przyrządach pomiarowych.
  - LPT[331]
- interfejsy typowo służących połączeniu komputera z wewnętrznymi modułami: ISA[332], PCI[333], PCI Express[334], ATA[335], SATA[336], SCSI[337], SAS[338], …
- Ethernet[339] - standard sieci komputerowych zyskujący coraz większe znaczenie w komunikacji urządzeń elektronicznych takich jak sterowniki, kontrolery itp, wygodny w użyciu, umożliwia stosowanie TCP/IP[340]. Niektóre z mikrokontrolerów posiadają warstwę MAC w sobie (co pozwala na dołączenie tylko prostszego układy warstwy fizycznej PHY), w przeciwnym razie konieczne jest dołączenie całego układu kontrolera sieciowego.

Istotnymi interfejsami komunikacyjnymi systemów elektronicznych są współcześnie sieci. Rozwiązania sieciowe mogą wykorzystywać jako warstwę sprzętową któryś z opisanych powyżej interfejsów szeregowych (w zasadzie poza UART i po części SPI wszystkie one są sieciami opisującymi 1 i 2 warstwę wg. modelu OSI) albo posiadać własny standard interfejsu (np. światłowodowy, bezprzewodowy, wykorzystujący okablowanie energetyczne). Obecnie popularność Ethernetu staje się na tyle istotna iż zaczyna on wypierać popularne dotychczas standardy sieci przemysłowych służących do komunikacji pomiędzy sterownikami (fieldbus[341]) takie jak: Interbus[342], Foundation Fieldbus[343], CC-Link[344]. Ponadto zauważalna jest migracja wielu protokołów tego typu sieci (standardów nie definiujących warstwy fizycznej i łącza danych) w stronę wykorzystania Ethernetu czy nawet TCP/IP, za przykłady mogą posłużyć tutaj:

- Modbus[345] - protokół wykorzystujący jako łącza transmisyjne UART lub Ethernet
- BACnet[346] - otwarty protokół komunikacyjny mogący korzystać m.in. z warstwy IP
- M-BUS[347] - dwu częściowa (format komunikatów i sposób transmisji przez sieć energetyczną) sieć do komunikacji z licznikami energii itp
- Profibus DP[348] - popularny standard sieci, wykorzystujący w warstwie fizycznej łącza RS485, optyczne lub transmisję po liniach energetycznych korzystający zarówno z mechanizmu odpytywania master-slave jak i wymiany znaczników pomiędzy masterami (token passing[349], tak jak w wymierającym już standardzie sieci komputerowych Token Ring[350]), aktualnym kierunkiem ewolucyjnym tego standardu jest oparta na Ethernecie sieć Profinet[351]
- S-bus[352] - protokół wykorzystujący jako łącza transmisyjne UART lub Ethernet
- LonWorks[353] - protokół komunikacyjny dla automatyki głównie inteligentnych budynków[354], który dzięki użyciu IP-852[355] może być przesyłany z użyciem sieci TCP/IP.

W tym miejscu należy też wspomnieć o protokołach używanych do udostępniania informacji przez różne systemy w sieciach TCP/IP takich jak:

- SNMP[356]
- OPC[357] (pomimo Windowsowych kożeni posiada także otwarte implementacje w pythonie działające także w innych systemach - OpenOPC for Python[358], PyOPC[359])

W specyficznych zastosowaniach (np. motoryzacja) można spotkać wiele innych rozwiązań w mniejszym bądź większym stopniu zbliżonych do tu wymienionych.

Zestawienie sieci czujnikowych

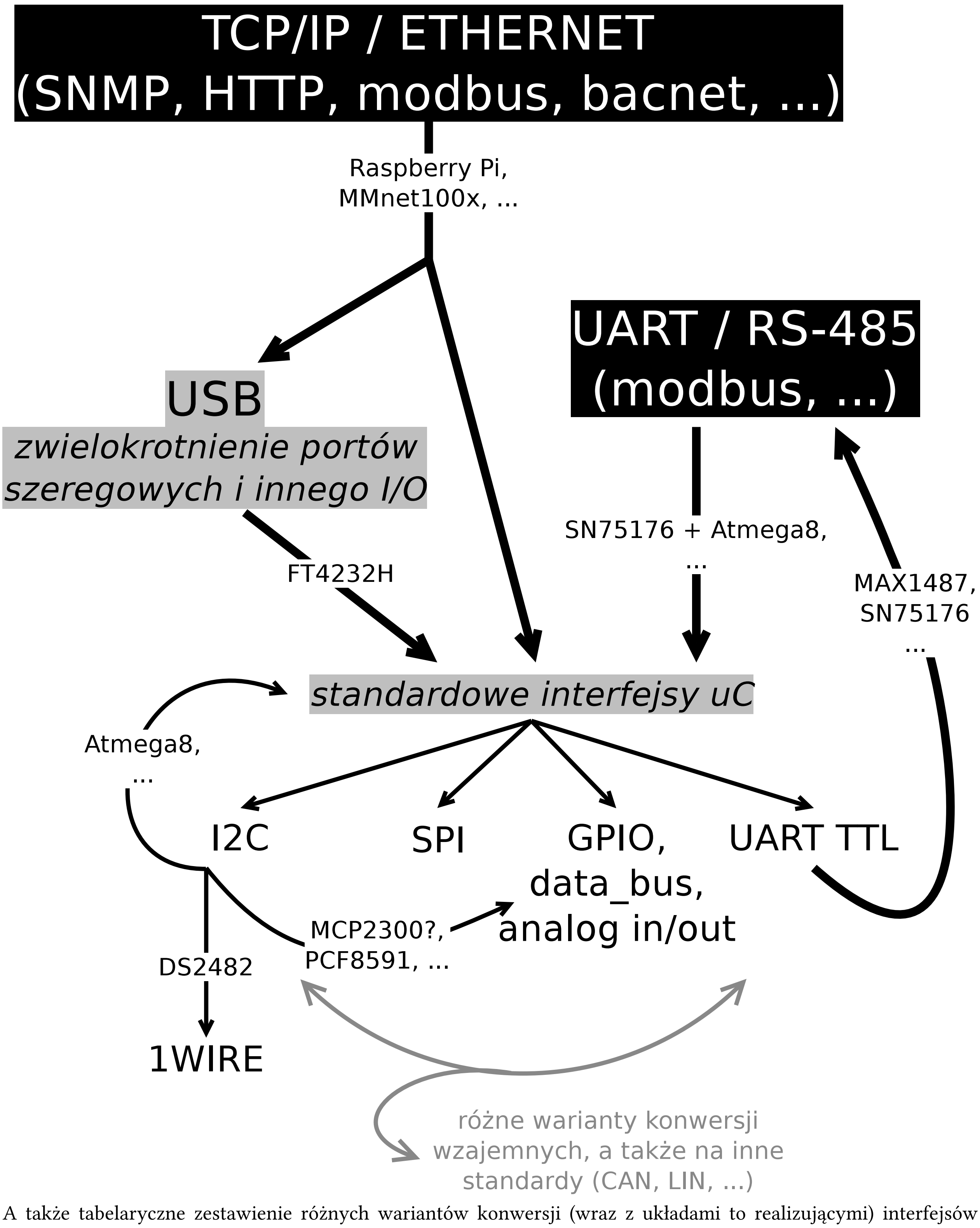
<table>
<tr><td>aplikacyjna
transportowa
sieciowa</td><td></td><td rowspan="2">inne / dane</td><td colspan="2">dane</td><td rowspan="2">BACnet MS/TP</td><td rowspan="2">P-Net[360]</td><td rowspan="2">Profibus[361]</td><td rowspan="2">inne / dane</td><td rowspan="2">BACnet PTP</td><td>M-Bus</td></tr>
<tr><td>łącza danych</td><td>format ramki komunikatu wraz z adresacją</td><td>Modbus[362] RTU</td><td>Modbus ASCII</td><td>IEC 60870</td></tr>
<tr><td>łącza danych</td><td>format przesyłu bajtu</td><td colspan="9">UART[363]</td></tr>
<tr><td>fizyczna</td><td>standard elektryczny</td><td colspan="5">RS-485[364]</td><td>inne</td><td colspan="2">RS-232[365]</td><td>M-Bus[366]</td></tr>
</table>

aplikacyjna transportowa sieciowa łącza danych	format ramki komunikatu wraz z adresacją	inne / dane	DeviceNet[367]	CANopen[368]	inne / dane	inne / dane	inne / dane	inne / dane	LonWorks	EIB / KNX
łącza danych fizyczna	format przesyłu bajtu standard elektryczny	CAN[369] (ISO 11898)			SPI[371]	I2C[370] / TWI	1Wire	USB		

aplikacyjna		inne / dane	inne	SNMP	Modbus TCP	BacNet IP	KNXnet/IP	Profinet
transportowa sieciowa					TCP / IP			
łącza danych	format ramki komunikatu wraz z adresacją							
łącza danych fizyczna	format przesyłu bajtu standard elektryczny		Ethernet[372] / inne				Ethernet[373]	

KONWERSJE

Diagram prezentujący kierunki konwersji typowych interfejsów w systemie automatyki opartym na mikrokontrolerach:



A także tabelaryczne zestawienie różnych wariantów konwersji (wraz z układami to realizującymi) interfejsów typowych dla elektroniki (typowych interfejsów mikrokontrolera) oraz popularnych sieci czujnikowych:

konwersja z (kolumny) / na (wiersze):		I2C	UART TTL	GPIO, data_bus	SPI
I2C / TWI / SMBus					CP2120
UART	TTL	MAX3107, SC16IS7?0		SC26C92	MAX310?, SC16IS7?0
UART	RS232	MAX232, …			
UART	RS485	SN75176, MAX485, …			
GPIO, data_bus		, PCF8574, PCA9554			MCP23S0?
Analog input/output					
SPI		SC18IS602			
1Wire			DS2480		
CAN	warstwa fizyczna				MCP2515
CAN	warstwa łącza danych			MCP2515	
LIN	warstwa fizyczna		MCP2003		
LIN	warstwa łącza danych				
Mbus / Metter-Bus	warstwa fizyczna		MSP430		

Zobacz także: SNMP-BACnet gateway, BACnet Stack[374], libmodbus[375].

Elementy teorii układów logicznych

Podstawowym podziałem jest podział na układy kombinacyjne[376] (stan ich zależy tylko od aktualnego wejścia) oraz sekwencyjne[377] (stan ich zależy od aktualnego wejścia oraz historii układu). Na potrzeby teoretycznej analizy układów kombinacyjnych często stosuje się ich opis z wykorzystaniem tabeli gdzie kolejne wiersze odpowiadają różnym wektorom pobudzającym (kodowanym w NKB[378]) a kolumny pokazują sygnały na wyjściach układu.

Minimalizacja - metoda tablic Karnaugha

Aby uzyskać wyrażenie na opisywaną przez tablicę funkcję wypisujemy w oparciu o zakreślenia:

1. jedynek - **sumę ilorazów zadanych przez zmienne** niezmieniające się w ramach pętelki,
2. zer - **iloraz sum zadanych przez zanegowane zmienne** niezmieniające się w ramach pętelki.

Możemy także wypisać funkcję zaprzeczoną - w tym celu dla obejmowania pętelkami jedynek postępujemy jak w pkt. 2 a dla zer jak w pkt. 1 (zamieniamy rolami zera z jedynkami w wartościach funkcji podanych w tablicy).

Warto także zapoznać się z komputerowym systemem minimalizacji funkcji logicznych espresso[379]. Zamieszam także skrypt do generowania plików PLA na podstawie wyrażenia boolowskiego - expr2pla.sh

Ekspansja

macierz blokująca

Macierz blokująca dla kostki k_i (i funkcji f) jest to macierz utworzona z macierzy wektorów r_i dla których f ma wartość 0, poprzez zanegowanie kolumn o indeksach j dla których $k_{ij} = 1$.

POKRYCIE KOLUMNOWE

Jest to zbiór kolumn L_j takich że w każdym wierszu co najmniej jedna z kolumn ma wartość 1. Minimalnym pokryciem nazywamy takie pokrycie, dla którego usunięcie z niego jakiejkolwiek kolumny sprawiłoby że przestaje być ono pokryciem. Możemy także wprowadzić relację mniejszości pokrycia równoważną relacji mniejszości ilości kolumn składających się na pokrycie.

Minimalne pokrycia obliczamy poprzez minimalizację wyrażenia: $\bigcap_i \left(\bigcup_{j:b_{ij}=1} L_j\right)$. Wyrażenie doprowadzamy do postaci sumy iloczynów, reprezentującej wszystkie minimalne pokrycia kolumnowe zadanej macierzy B.

IMPLIKANTY PROSTE

Implikanty uzyskujemy poprzez wybór z wektora k_i wartości k_{ij} dla kolumn wchodzących w skład minimalnego pokrycia kolumnowego, gdy $k_{ij} = 1$ to do implikantu wchodzi x_j w przeciwnym wypadku ($k_{ij} = 0$) wchodzi wartość zanegowana $\bar{x}_j$.

SELEKCJA NAJLEPSZYCH IMPLIKANTÓW

Najpierw ustalamy które kostki k_i pokrywają dany implikant I_j (kostka pokrywa implikant gdy dla każdej wartości z kostki implikant ma wartość z nią zgodną lub dowolną). Następnie tworzymy macierz P o kolumnach zawierającą informację o tym pokrywaniu (gdy k_j pokrywa I_j to $p_{ij} = 1$, w przeciwnym wypadku jest 0) i obliczamy minimalne pokrycie kolumnowe. Suma implikantów z minimalnego pokrycia kolumnowego stanowi minimalną formułę dla funkcji początkowych.

Redukcja argumentów

PODZIAŁ

Podziałem nazywamy system zbiorów o rozłącznych blokach, których suma daje cały zbiór. Podział A jest nie większy od B gdy każdy blok A zawiera się w jakimś bloku B.

Iloczyn podziałów (największy podział nie większy od każdego z wchodzących w skład iloczynu) obliczamy grupując elementy w bloki tak aby każdy z bloków zawierał się w jakimś bloku każdego z składników oraz aby były to bloki możliwie największe.

$A = \{\overline{1, 2, 3}, \overline{4}\}$
$B = \{\overline{1, 2}, \overline{3, 4}\}$
$A \cdot B = \{\overline{1, 2}, \overline{3}, \overline{4}\}$

Ilorazem podziałów nazywamy podział (dzielony) z blokami złożonymi z powiększonych elementów (stanowiących bloki podziału przez który dzielimy).

$A = \{\overline{1, 2, 3}, \overline{4}\}$
$B = \{\overline{1, 2}, \overline{3}, \overline{4}\}$
$A/B = \{\overline{(1, 2), (3)}, \overline{(4)}\}$

PROCEDURA REDUKCJI

Obliczamy iloczyn podziałów względem **zmiennych niezbędnych** (zmiennych na których różnią się dwa wektory dla których funkcja ma różne wartości i nie mają one innych różnych pozycji) i dzielimy go przez podział względem wyniku zwracanego przez funkcję. Następnie znajdujemy pozycje na których różnią się wektory o numerach należących do różnych elementów składających się na obliczony podział. (dla powyższego przykładu będą to różnice pomiędzy wektorem 3 a 1 i 3 a 2). Ostatecznie obliczamy iloczyn sum tych zmiennych (na którym różnią się wektory). Po przekształceniu go na sumę iloczynów każdy składnik tej sumy wraz z zmiennymi niezbędnymi stanowi zestaw argumentów, oczywiście wybieramy ten najmniejszy … .

Dekompozycja

Czyli (rozsądny) podział na podfunkcje (tak aby każda z nich zależała od mniejszej liczby zmiennych niż wejściowa)

Tworzymy macierz gdzie kolumny odpowiadają pewnemu podzbiorowi argumentów wejściowych, a wiersze reszcie zbioru argumentów wejściowych. Wyszukujemy w niej kolumny zgodne (o takich samych wartościach lub akceptujące takie same wartości). Dokonujemy optymalnego sklejenia kolumn zgodnych otrzymując w ten sposób funkcję g o ilości argumentów nie większej niż wielkość podzbioru przeznaczonego na opis kolumn oraz ilości wyjść odpowiadającej logarytmowi ilości kolumn po sklejeniu przybliżonemu w górę.

Obliczanie maksymalnych klas zgodnych metodą par zgodnych

Para zgodna = dwie kolumny zgodne ze sobą.

W oparciu o indeksy elementów wchodzących w skład par zgodnych tworzymy zbiory R_i. R_i jest zbiorem takich j że (i, j) jest parą zgodną oraz $i > j$.

Mając obliczone R_J generujemy rodzinę maksymalnych klas zgodnych wg następujących reguł:

1. jeżeli R_k jest zbiorem pustym to rodzinę MKZ powiększamy o KZ k
2. jeżeli R_k przecięty z KZ jest pusty to bez zmian
3. jeżeli R_k przecięty z KZ jest nie pusty to to przecięcie powiększamy o k i zastępujemy nim daną KZ

Punkty 2 i 3 powtarzamy dla każdej klasy zgodnej w rodzinie. Jeżeli jakaś KZ zawiera się w innej to o niej zapominamy.

Obliczanie maksymalnych klas zgodnych metodą par sprzecznych

Pary sprzeczne to wszystkie pary które nie są zgodne. Poprzez zbudowanie wyrażenia boolowskiego typu iloczyn sum (gdzie każda suma jest zbudowana z jednej pary sprzecznej) i przekształcenie go do postaci sumy iloczynów, uzyskujemy rodzinę MKZ (każdy iloczyn wchodzący do tej sumy reprezentuje MKZ).

Rodzaje automatów

Automat jest opisem matematycznym dużej liczby układów cyfrowych, ułatwiających ich konstruowanie. Opiera się na wyróżnieniu dyskretnych stanów i czynników powodujących zmianę stanu. Zobacz też: Automat skończony[380], Deterministyczny automat skończony[381].

Moor'a

Automat Moore'a[382] charakteryzuje się powiązaniem wyjść wyłącznie z stanem wewnętrznym automatu, czyli automat będący w określonym stanie zawsze ma takie samo wyjście. Automat możemy opisać przy pomocy grafu[383], gdzie wierzchołkami są stany z powiązanymi z nimi wyjściami, a krawędziami są przejścia pomiędzy stanami (wraz z powodującym je sygnałem). Inną formą prezentacji jest tabela przejść/wyjść, gdzie dla każdego stanu podane są stany następne do których automat przechodzi pod wpływem danego wektora oraz wyjście związane z aktualnym stanem:

stan obecny	wektor wejściowy 1	wektor wejściowy 2	...	wyjście
A	B	A	...	10
B	C	D	...	11
...				

Mealy'ego

Automat Mealy'ego[384] charakteryzuje się powiązaniem wyjść z stanem wewnętrznym automatu i jego wejściami. Automat możemy opisać przy pomocy grafu[385], gdzie wierzchołkami są stany, a krawędziami są przejścia pomiędzy stanami (wraz z powodującym je sygnałem i związanym z tym wyjściem automatu). Inną formą prezentacji jest tabela przejść/wyjść, gdzie dla każdego stanu podane są stany następne do których automat przechodzi pod wpływem danego wektora oraz wyjścia związane z aktualnym stanem i podanym wektorem wejściowym:

stan obecny	wektor wejściowy 1	wektor wejściowy 2	...
A	B/10	A/11	...
B	C/01	D/11	...
...			

(a)synchroniczne

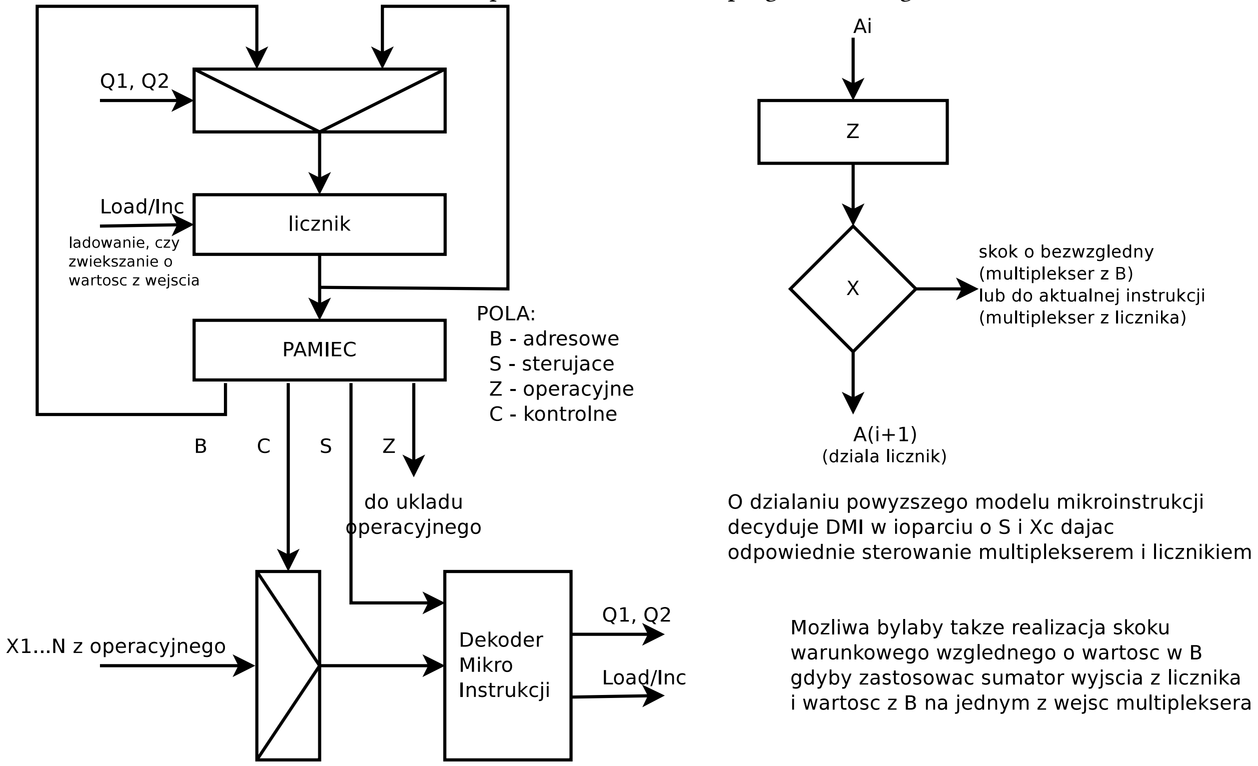
Automatem synchronicznym nazywa się automat który działa zgodnie z zewnętrznym sygnałem taktującym, którego zbocze powoduje dokonanie przejścia stanu w automacie. Automat asynchroniczny stan swój zmienia natychmiast po zmianie wektora wejściowego (dowolnego z jego bitów). W tym wypadku należy mieć na szczególnej uwadze problemy związane z tym że nigdy nie możemy oczekiwać równoczesnej zmiany dwóch sygnałów co prowadzi do tego że np. przejście ze stanu kodowanego jako 11 na 00 będzie przez stan 10 albo 01. Prowadzi to do tak zwanych wyścigów. jeżeli automat może utknąć w którymś z tych stanów są to tzw. wyścigi krytyczne i należy je eliminować.

Synteza automatów

Synteza automatu sprowadza się do następujących kroków:

- opis w postaci grafu
- konwersja grafu na tablicę przejść/wyjść
- kodowanie (numeracja stanów)
- zapisanie tablicy przejść/wyjść w postaci tablicy Karnaugha
- podział na tyle tablic ile bitów ma kod stanu (na każdy przerzutnik trzymający jeden bit numeru stanu automatu jedna tablica) - każda tablica jako wejścia przyjmuje poprzedni wektor stanu i aktualny wektor wejść
- jeżeli używamy przerzutników innych niż D to należy dokonać konwersji tych tablic poprzez tablice dla poszczególnych typów przerzutników
- minimalizacja tablic opisujących logikę podłączoną do danego przerzutnika
- zdefiniowanie logi przekształcającej wektor stanu (i ew. wektor wejściowy jeżeli automat typu Mealy'ego) na wektor wyjściowy

Automat można zrealizować także w postaci układu mikro-programowalnego:



Minimalizacja automatów (liczby stanów) - synchroniczne

- tworzymy tablicę trójkątną (kolumny 1 -> N-1, wiersze 2-> N) w której zaznaczamy dla każdej pary stanów czy jest ona:
  - zgodna (dla każdego wejścia stany mają niesprzeczne wyjścia i niesprzeczne stany następne)
  - warunkowo zgodna (dla każdego wejścia stany mają niesprzeczne wyjścia i niesprzeczne stany następne, pod warunkiem zgodności innych stanów) - wtedy zapisujemy warunki zgodności (jest to iloczyn logiczny)
  - sprzeczna (dla jakiegoś wektora wejściowego wyjścia są sprzeczne)
- po wypełnieniu sprawdzamy dla każdej pary sprzecznej czy nie jest warunkiem w innej parze - jeżeli jest to tamta para też sprzeczna, powtarzamy to dla nowo powstałych par sprzecznych, aż dojdziemy do sytuacji że nie ma już nowo powstałych par sprzecznych
- pozostałe pary (zgodne lub zgodne warunkowo) są parami zgodnymi
- obliczamy MKZ (można jak opisano powyżej, ale najprościej poprzez grupowanie 3 par w trójkę, 4 trójek w czwórkę itd)
- wybieramy takie MKZ aby spełnić warunek pokrycia (wszystkie stany muszą być w zbiorze wynikowym) i sprawdzamy czy jakiś stan z jednej grupy nie prowadzi do stanów z innej grupy
- w tym celu sporządzamy tabelkę gdzie w kolumnach mamy kolejne MKZ, a w wierszach mamy ich reakcję na kolejne wektory wejściowe

Minimalizacja automatów (liczby stanów) - asynchroniczne

- łączymy stany pseudo równoważne (mające stany stabilne w jednej kolumnie, niesprzeczne wyjścia oraz niesprzeczne stany następne), w tym etapie rozważamy zgodność warunkową
- tworzymy graf zgodności (warunkiem jest niesprzeczność stanów następnych), stany o sprzecznych wyjściach umieszczamy, ale łączymy je linią przerywaną
- przy automacie Mealy'ego możemy grupować stany o sprzecznych wyjściach (połączone linią przerywaną), robimy wtedy także tablicę wyjść:
  - przepisujemy wyjścia stanów stabilnych

    - przepisujemy wyjścia stanów niestabilnych (gdy dane wyjście nie jest wyjściem stabilnego z bieżącego wejścia to kreska)
    - uzupełniamy kreski
- kodujemy tak aby uniknąć wyścigów krytycznych - przejścia przez stany niestabilne i/lub dodajemy dodatkowe stany

Komputery

Komputer[386] jest elektronicznym urządzeniem służącym do przetwarzania informacji. Działanie jego opiera się na przetwarzaniu ciągu rozkazów (wraz z argumentami którymi są przetwarzane dane). W przypadku dominującej obecnie architektury[387] von Neumanna[388] (w odróżnieniu od harwardzkiej[389] stosowanej bardzo często w mikrokontrolerach[390]) zarówno ciąg instrukcji (program) jak i dane przechowywane są w wspólnej pamięci.

Głównym parametrem klasyfikującym systemy komputerowe jest architektura procesora[391]. Determinuje ona m.in. zestaw obsługiwanych instrukcji i wielkość przestrzeni adresowej, w związku z tym programy kompilowane do kodu maszynowego w celu uruchomienia na innej, niekompatybilnej architekturze wymagają rekompilacji. Do najistotniejszych obecnie grup architektur należy zaliczyć:

- x86[392] (IA-32[393] / AMD64[394]) - aktualnie najpopularniejsza w komputerach domowych / biurowych, konsolach do gier (ósma generacja[395]) i serwerach
- ARM[396] - aktualnie najpopularniejsza w telefonach komórkowych, często spotykana w różnych systemach wbudowanych
- MIPS[397] - często spotykana w routerach, tunerach DVB, rejestratorach, itp:
- Power[398] (PowerPC[399], POWER8[400], Cell[401], OpenPOWER[402]) - często spotykana w systemach serwerowych i konsolach do gier (głównie siódma generacja[403])
- SPARC[404] (OpenSPARC[405], LEON[406]) - często spotykana w systemach serwerowych

W ramach poszczególnych grup często zachowywana jest zgodność wsteczna (jest tak np. w przypadku rodziny x86), ale nie zawsze musi tak być. Należy zwrócić uwagę na różnice pomiędzy poszczególnymi wersjami architektury w ramach rodziny (rozszerzenia zestawu instrukcji), bitowość (32 lub 64 bity), warianty big-endian[407]/little-endian[408] (szczególnie dla MIPS i ARM), stosowanie koprocesorów (np. jednostką zmiennoprzecinkową[409] VFP w ARM). Zobacz także: Debian Multiarch Tuples[410]

Budowa - płyta główna, procesor, pamięć, karty, ...

Płyta główna[411] zawiera podstawowe elementy urządzenia (w tym wypadku komputera), umożliwia też komunikację wszystkim pozostałym podzespołom i modułom. Klasyczny, ogólny (generyczny) schemat blokowy płytywy głównej[412] zawiera 4 główne elementy:

- procesor[413] (CPU)
- mostek północny[414] połączony bezpośrednio z procesorem
- pamięć[415] (RAM[416]) podłączona do kontrolera pamięci[417] (klasycznie umieszczanego w mostku północnym, współcześnie najczęściej bezpośrednio w procesorze)
- mostek południowy[418] połączony z mostkiem północym

Z kolejnymi generacjami procesorów zmianie ulegały technologie wykonywania poszczególnych połączeń a także sam schemat (w wielu współczesnych konstrukcjach mostek północny nie występuje - pamięć podłączana jest bezpośrednio do procesora, a szybkie szyny systemewe do mostka południowego lub także bezpośrednio do procesora). Ogólnie zauważalne jest także odchodzenie od klasycznych magistral na rzecz łączy typu punkt-punkt (ew. magistral hub-owanych) oraz odchodznie od interfejsów równoległych na rzecz szeregowych. Aktualnie najistotniejszymi interfejsami są:

- HyperTransport[419], QPI[420] - łącza punkt-punkt pomiędzy procesorem a mostkiemi (następcy FSB[421]) oraz pomiędzy procesorami w systemach wieloprocesorowych[422]
- DMI[423] - łacze pomiędzy mostkiem północnym a południowym lub (wraz z FDI[424]) procesorem a mostkiem południowym
- PCIe[425] - szeregowe łącze typu punkt-punkt przeznaczone głównie do komunikacji z kartami rozszerzeń

— SATA[426], SAS[427] - szeregowe łącza punkt-punkt przeznaczone do komunikacji z urządzeniami pamięci masowej, SATA stosowane są powszechnie w rozwiązaniach domowych i biurowych, SAS niemal wyłącznie w rozwiązaniach serwerowych, interfejsy są ze sobą częściowo kompatybilne (kontroler SAS obsługuje dyski SATA)
— USB[428] - magistrala szeregowa, budowana w oparciu o hub-y[429], przeznaczona głównie do podłączania peryferiów zewnętrznych
— DVI[430], HDMI[431], DisplayPort[432] - interfejsy cyfrowego audio-wideo, pod względem cyfrowego sygnału wideo jest zgodne z DVI-D / DVI-I jest zgodne z HDMI (HDMI posiada dodatkowo kanał audio, obsługuje inne przestrzenie kolorów niż RGB, od wersji 1.4 może posiadać także kanał Ethernet; DVI w wariantach DVI-I i DVI-A posiada analogowy sygnał video zgodny z VGA[433])

Nadal można spotkać się z (wymierającymi już) interfejsami takimi jak: PCI[434]IEEE 1394[435] (FireWire), RS232[436] (port szeregowy[437] COM), PS/2[438], LPT[439] (port równoległy[440]), ATA[441] (IDE, PATA, ATAPI[442], EIDE). Natomiast wśród historycznych interfejsów należy wspomnieć o: ISA[443] (EISA[444], VESA Local Bus[445]), PCI-X[446] (poprawiona, szybsza, 64 bitowa, kompatybilna wstecz wersja PCI), PCMCIA[447] (pierwotnie "PC Card 16" - magistrala ISA, później "CardBus" - magistrala PCI), AGP[448], SCSI[449]. Zobacz także: porównanie prędkości różnych interfejsów[450].

Wiele z współczesnych (podobnnie jak i historycznych) interfejsów posiada różne warianty zarówno pod względem generacji / prędkości (typowo kompatybilnych wstecznie) jak i pod względem wykonania mechanicznego (zewnętrzne/wewnętrze, o różnym stopniu miniaturyzacji - w wielu wypadkach jest ich aż za wiele). Istnieją także złącza obsługujące kilka różnych interfejsów w ramach jednego gniazda - np. M.2[451] (w zależności od wariantu różne zestawy: PCIe, SATA, USB, DisplayPort, PCM/I²S[452], I²C[453]/SMBus[454], SIM[455]), SATA Express[456] (SATA + PCIe), eSATAp[457] (SATA + USB), ExpressCard[458] (PCIe + USB), Thunderbolt[459] (PCIe / DisplayPort).

Kontrolery poszczególnych interfejsów pamięci masowych, interfejsów zewnętrznych, a także inne układy (kontrolery macierzy dyskowych RAID[460], dźwiękowe[461], karty sieciowe[462], karty graficzne[463], ...) mogą być w postaci osobnych modułów (kart rozszerzeń) lub mogą być zintegrowane z płytą główną. W tym drugim przypadku często przynajmniej część ich funkcjonalności zawarta jest w mostku południowym, jednak mogą też być umieszczone w postaci osobnych układów podłączonych do jakiejś szyny obsługiwanej przez któryś z mostków lub procesor, albo nawet zintegrowane z procesorem (jak ma to niekiedy miejsce w przypadku GPU[464]). Praktycznie wszystkie współczesne płyty główne posiadają zintegrowane karty sieciowe Ethernet[465] oraz układy dźwiękowe HD (obsługujące wysokiej jakości dźwięk przestrzenny 7.1). Płyta też zawsze zawiera m.in. podstarzały już Basic Input/Output System (BIOS[466]) lub nowszy jego zamiennik UEFI[467] odpowiedzialny za start urządzenia, zegar czasu rzeczywistego, kontroler przerwań (informuje w rozsądny sposób - bez konieczności ciągłego odpytywania procesor o wydarzeniach w systemie - naciśnięcie klawisza, sygnały z urządzeń - np. koniec jakiejś operacji ...)

Zobacz w Sieci: Komputer i Sieci na Wydziale Fizyki UW (archive.org)[468], Hardware Book[469], AV info[470], Podstawowe komponenty komputera[471].

Inne elementy systemu komputerowego

Oprócz opisanych powyżej fundamentalnych składników systemu komputerowego oraz najistotniejszych interfejsów w skład systemu wchodzą inne urządzenia takie jak:

— Human interface device[472] - Input device[473]: mysz[474], klawiatura[475], niekiedy także: Trackball[476] / TrackPoint[477] / tablet[478], dżojstik[479] / GamePad[480], itp
— Human interface device - Output device[481]: monitor[482] (dominuje technologia LCD coraz częściej z podświetlaniem LEDowym[483]), projektor, drukarka[484] (niekiedy z wbudowanym skanerem[485]), ploter[486]
— multimedialne: kamery, aparaty cyfrowe, głośniki[487], słuchawki[488] (z mikrofonem[489] lub bez), tunery[490] radiowo[491]-telewizyjne[492] (DVB)
— napędy dysków optycznych[493] (DVD[494]), nagrywarki[495] (CD-R[496], CD-RW[497], DVD+[498]/-R[499], DVD+[500]/-RW[501]), czytniki kart pamięci flash[502] (dominującym standardem jest Secure Digital[503] / microSD[504] (TransFlash))
— modemy[505]

Uwagi końcowe

Osiągnięto chyba już osiągnięto kres podnoszenia zegara na rzecz tworzenia systemów wielordzeniowych. Standardem stała się także zgodna z x86 64-bitowa architektura procesorów opracowana przez AMD (AMD64[506]). Istotny jest też wzrost pamięci podręcznej, która będąc znacznie szybsza od RAMu[507] (i tak dużo szybszego od dysków twardych) przechowuje najbardziej obecnie potrzebne dane i rozkazy (do ustalenia które używane są różne rodzaje kolejek - idealnie byłoby usuwać to co najdłużej będzie niepotrzebne, ale tak się nie da - nie znamy przyszłości), tym samym przyspieszając przetwarzanie. Na rynku komputerowym jakoś nie mogą przebić się napędy blue-ray[508].

Należy niestety stwierdzić iż obecnie brakuje naprawdę wielkopojemnych (rzędu 40% wilkości przeciętnego współczesnego dysku twardego) naśników wymiennych, co rodzi spore problemy/niedogodności przy archiwizacji danych.

Więcej bardziej konkretnych (prędkości, etc.) informacji o poszczególnych interfejsach znaleźć można np. w Wikipedii[509]. Warto zapoznać się także m.in. z dokumentacją układu umożliwiającego konwersję sygnału VGA (składowe RGB) na sygnał Composite video[510] lub S-Video[511] - AD725[512]. Zobacz także: Jack plug[513], Registered jack[514], Modular connector[515], Bluetooth[516], IrDA[517], S/PDIF[518] / TOSLINK[519]; systemy emisji sygnału telewizyjnego[520], IEC_169-2[521]; Kategoria:Sprzęt_komputerowy[522], Kategoria:Komputery[523], Kategoria:Magistrale[524], Kategoria:Pamięci_komputerowe[525], Kategoria:Nośniki_danych[526] (Kategoria:Dyski_optyczne[527], Kategoria:Dyski_magnetyczne[528]).

Rozwiązania serwerowe

Komputery używane w profesjonalnych zastosowaniach pod względem koncepcyjnym często nie różnią się znacznie od zwykłych PCtów. Cechuje je jednak:

- stosowanie rozwiązań o większej wydajności,
- stosowanie nowoczesnych rozwiązań które być może w przyszłości trafią do komputerów domowych (ale nie koniecznie),
- stosowanie rozwiązań zwiększających niezawodność (redundacja[529], hot-plug[530]) i ułatwiających zarządzanie (hot-plug, zarządzanie out-of-band[531]),
- nierzadko lepsza jakość i zawsze zdecydowanie większa cena.

Urządzenia (komputery, szpręt sieciowy, macierze dyskowe, itd.) montowane są w szafach lub stojakach rackowych[532] o zestandaryzowanych wymiarach (wysokość urządzeń określana jest w U[533], rozstaw stelaży 19" szerokość szafy 60 lub 80 cm, różne głębokości - do 120 cm).

Redundancja i Hot plugging

Wiele komponentów montowanych jest nadmiarowo - np. instalowane i podłączane są dwa niezależne zasilacze z których każdy jest w stanie zasilić cały system, podobnie z wentylatorami. Stosowane są macierze dyskowe RAID budowane z dysków wewnętrznych celem zapewnienia odporności na awarię conajmniej jednego z dysków. Niezależnie kablowane są dwa interfejsy sieciowe w celu budowy redundantnej sieci (odpornej np. na awarię jednego ze switchy). Możliwa jest wymiana zasilaczy, wentylatorów, dysków twardych, kart rozszerzeń PCIe, itd. w trakcie normalnej pracy systemu.

Zarządzanie out-of-band

Przy zarządzaniu dziesiątkami, a często nawet setkami serwerów istotne są narzędzia pozwalające robić to zdalnie (i to nie tylko jak SSH na poziomie działającego systemu operacyjnego). Większość serwerów posiada też mniej lub bardziej udane układy zarządzania out-off-band (na ogół jest to mały komputerek działający w oparciu o ARM i linux) umożliwiające dostęp do konsoli szeregowej głównego sytemu (SOL[534]), przekierowanie konsoli graficznej, sterowanie zasilaniem serwera (zdalny restet), itd. Układ ten może używać własnego interfejsu sieciowego lub współdzielić go z głównym systemem. Podstawowe funkcjonalności takiego sposobu zarządzania standaryzuje IPMI[535], niestety nie starndaryzuje ono (dostępnych w bardziej zaawansowanych rozwiązaniach) KVM over IP[536] (które z kolei na ogół nie stosuje standardowego VNC[537]) i [538]. Zobacz także: IPMI.

Wysoka gęstość

Wśród rozwiązań serwerowych istotne miejsce zajmują serwery kasetowe[539] pozwalające na zwiększenie upakowania systemów komputerowych. Puszka (chassis) w której montowane są serwery oprócz zasilania i chłodzenia zapewnia im także interfejsy komunikacyjne (od tyłu umieszczane są switch lub moduły passthrough[540], półka zapewnia do kilku nawet różnych interfejsów dla każdego z serwerów). W niektórych przypadkach pozwala także na bezpośrednie montowanie w sobie np. macierzy dyskowych. Sam serwer kasetowy może być wyłącznie nodem obliczeniowym (bez dysku i miejsca na karty rozszereń), bądź nawet pełnoprawnym serwerem z redundantnymi dyskami hotswap i możliwością monatrzu kart rozszerzeń.

Zbliżonym do serwerów kasetowych jest rozwiązanie polegające na umieszczeniu kilku serwerów w wspólnej obudowie z backplane-m[541] PCIe i/lub SAS zapewniającym dostęp do wspólnych modułów I/O lub dysków. W odróżnieniu od serwerów kasetowych rozwiązania te nie zapewniają wspólnego zarządzania, itp.

Pamięci masowe

Omawiając systemy serwerowe nie można pominąć także macierzy dyskowych umożliwiających umieszczanie zestawów dysków (łączonych w celu zwiększenia niezawodności oraz pojemności) w osobnych obudowach i bezpośrednie (DAS[542]) dołączanie ich do kilku serwerów jednocześnie. Taka macierz dyskowa RAID >0 z redundantnym układem zasilania oraz kontrolera, umożliwiająca wymianę wszystkich podzespołów bez wyłączania lub restartu systemu (pełny hot plug[543]), może być bezpośrednio podłączana do np. dwóch serwerów które mogą wzajemnie przejmować swoje funkcje.

Większe macierze mogą bezpośrednio udostępniać zasoby plikowe w ramach NAS[544] lub zasoby blokowe w ramach SAN[545]. Sieci te budowane są w opraciu o sieci FC[546], IB[547] lub szybkie sieci Ethernet[548] (dla dostępów blokowych protokoły: iSCSI[549], FCoE[550], dla plikowych: NFS[551], SMB[552]).

Warunki środowiskowe

Ze względu na znaczne zurzycie enegii na niewielkiej przestrzeni związane z dużym upakowaniem systyemów serwerowych istotnym aspektem jest chłodzenie systemów serwerowychi. Obecnie odchodzi się od potrzeby zapewnienia do tego celu powietrza o bardzo stabilnych parametrach i dość niskiej temperaturze (kilkanaście stopni cencjusza na wlocie) na rzecz chłodzenia wysokotemperaturowego (około 25 stC na wlocie) lub nawet bezpośredniego chłodzenia wodnego[553]. Polega ono na wprowadzaniu dość ciepłej cieczy bezpośrednio na silnie grzejące się elementy komputera celem ich schłodzenia, dzięki bezpośredniemu odbiorowi ciepła z bardzo gorących elementów takich jak procesor czy też pamięć nie ma potrzeby chłodzenia tej cieczy w inny sposób niż w suchych chłodnicach (współczesne komputery pozwalają na chłodzenie cieczą o temperaturze na wejściu do komputera powyżej 44 stC). Należy jednak pamietac iż nawet w takim wypadku konieczne jest wychłodzenie[554] przy pomocy powietrza pozostałych komponentów komputera. Na koniec należy wspomnieć o specjalnych wykonaniach zarówno serwerów jak i komputerów przenośnych związanych z (mniej przyjaznymi od klimatyzowanej serwerowni) warunkami w których muszą pracować - np.: komputery wojskowe[555], rugged computer[556].

Wprowadzenie do systemów operacyjnych

System operacyjny[557] jest oprogramowaniem odpowiedzialnym za zarządzanie zasobami systemu komputerowego (sprzętem, ale nie tylko) oraz uruchomionymi na nim aplikacjami. Do najistotniejszych zadań systemu operacyjnego zalicza się podział czasu procesora i szeregowanie zadań[558] oraz zarządzanie pamięcią - w szczególności obsługa pamięci wirtualnej[559], najczęściej z wykorzystaniem mechanizmu stronicowania[560].

Oprócz tego system zajmuje się także zarządzaniem plikami, wejściem/wyjściem (najczęściej jest ono realizowane w oparciu o przerwania[561] (IRQ), ale znane są także modele programowego we/wy polegającego na aktywnym czekaniu), obsługą urządzeń (wejście/wyjście, sterowniki, dostęp), obsługą sieci (stos protokołów sieciowych), itd. Część zadań realizowana jest z minimalnym udziałem procesora (a więc także i systemu) jest to na przykład transfer danych w trybie DMA polegający na tym iż dane kopiowane są całymi blokami bez udziału procesora do/z pamięci (system zajmuje się tylko inicjacją transmisji). Należy tu jednocześnie zaznaczyć iż w przypadku nie stosowania tej technologii dane też kopiowane są pomiędzy dyskiem a procesorem całymi blokami (minimum sektor) gdyż dysk (w odróżnieniu od pamięci operacyjnej) nie jest bezpośrednio dostępny dla procesora.

Współczesne systemy korzystają z co najmniej dwóch poziomów pracy - uprzywilejowanego poziomu "nadzorcy" w którym działa jądro systemu operacyjnego oraz trybu użytkownika (zobacz też tryb rzeczywisty[562] i tryb chroniony[563]). Operacje I/O muszą odbywać się w trybie uprzywilejowanym. Również pamięć posiada obszar chroniony, w którym umieszczany jest m.in. tablica wektorów przerwań[564] (inaczej zmiana adresu w tym wektorze mogłaby doprowadzić do przejęcia systemu w trybie uprzywilejowanym).i

Procesy i szerwegowanie zadań

Istotną rolą systemu operacyjnego w zarządzaniu procesami[565] (obok czynności administracyjnych jak ich tworzenie powielanie, usuwanie, czy też wstrzymywanie itp) jest zapewnienie ochrony pamięci (każdy proces może

pisać po swojej i ewentualnie współdzielonej gdy dostał do tego prawo) oraz procesora (przerwanie zegarowe powoduje wywołanie planisty, który ustala jaki proces dostanie następny kwant czasu procesora). Niektóre systemy wyróżniają obok procesów także wątki, które różnią się od nich współdzieloną (między wątkami jednego procesu) pamięcią i zasobami (np. otwartymi plikami). System operacyjny zapewnia także zestaw usług i funkcji (wywołań) systemowych zapewniających pośrednictwo między interfejsem trybu użytkownika a sprzętem.

Istotnym zadaniem systemu operacyjnego jest przeciwdziałaniem tzw. blokadom, czyli sytuacji gdy dwa lub więcej procesów blokują się wzajemnie w oczekiwaniu na zasoby (a ma zasób X, którego potrzebuje b aby zwolnić zasób Y, którego potrzebuje a do zwolnienia X). Realizowane to może być na kilka sposobów:

- zapobieganie blokadzie (czyli niedopuszczenie do zajścia warunków koniecznych) - np. poprzez konieczność deklarowania wszystkich zasobów na początku, zwalniania przydzielonych zasobów przez zgłoszeniem zapotrzebowania na następne
- unikanie blokady (czyli określamy maksymalne zapotrzebowanie i tak przydzielamy zasoby aby uniknąć zajścia blokady) - np. poprzez kontrolę czy po spełnieniu żądania dalej będziemy działać w stanie "bezpiecznym", tj takim że istnieje sekwencja (zwana bezpieczną) w której maksymalne zapotrzebowanie każdego procesu może być spełnione w oparciu o zasoby zwolnione przez procesy będące wcześniej w tej sekwencji oraz zasoby wolne, zobacz też: algorytm bankiera[566]
- wykrywanie i usuwanie blokady gdy do niej doszło

Planista procesora, czyli fragment systemu odpowiedzialny za przydzielanie procesora procesom, może pracować w trybie z wywłaszczaniem lub bez. W tym pierwszym wypadku proces otrzymuje kwant czasu procesora który może wykorzystać w całości (wtedy przejdzie z stanu wykonywania w stan gotowości) lub z niego wcześniej zrezygnować (gdy np. czeka na I/O, wtedy przejdzie z stanu wykonywania w stan oczekiwania). W drugim przypadku proces wykonuje swój kod do momentu aż sam odda procesor. Forma ta zbliżona jest do wykorzystywanej w szeregowaniu czasu rzeczywistego[567] - proces będzie wywłaszczony tylko przez proces o wyższym priorytecie i będzie to natychmiastowe (przy najbliższym przerwaniu zegarowym) - zobacz priorytety.c. Istnieje wiele algorytmów szeregowania takich jak:

- FCFS - pierwszy zgłoszony = pierwszy obsłużony
- SJF - najkrótszy zgłoszony będzie pierwszym wykonanym (wersja z wywłaszczaniem - SRTF - gdy nowy najkrótszy pozostały), algorytm raczej nie do zastosowania praktycznego - trzeba by przewidywać długość wykonania
- priorytetowe - zawsze o najwyższym priorytecie (jak wspomniałem wyżej wykorzystywane w systemach real-time[568]
- rotacyjne - każdy po kawałku, potem na koniec kolejki
- kolejki wielopoziomowe - system z priorytetami, podziałem czasu pomiędzy kolejki, przenoszeniem procesów między kolejkami, ...

Zarządzanie pamięcią

Drugą podstawową funkcją systemu operacyjnego, wspomnianą na początku, jest zarządzanie pamięcią. Zarządzanie pamięcią polega na odpowiednim mapowaniu adresów logicznych (używanych przez procesy) na adresy fizyczne (używane przez procesor), korzysta ono z wsparcia sprzętowego ze strony procesora. Jest to najczęściej realizowane w oparciu o wspomniany mechanizm stronicowania[569]. Polega to na podziale pamięci dostępnej pamięci fizycznej na jednakowe bloki zwane ramkami oraz podziale pamięci logicznej na jednakowe bloki (o tej całej wielkości co ramki) zwane stronami. Strony które są wykorzystywane przez program są mapowane na dowolne ramki pamięci fizycznej (w przypadku gdy dana strona nie zamapowania - w zależności od okoliczności błąd braku strony lub błąd ochrony strony). Rozwiązuje to problem fragmentacji zewnętrznej, polegającej na braku spójnego obszaru pamięci o żądanej długości pomimo iż łączna ilość wolnej pamięci jest dostateczna, jednak nie rozwiązuje problemu fragmentacji wewnętrznej, polegającej na przydzielaniu zbyt dużych fragmentów pamięci dla procesu (a wręcz można powiedzieć że go pogłębia). Mechanizm ten wymaga trzymania tablicy wolnych ramek, tablicy stron dla każdego procesu (zawierającej przypisania mapowań stron danego procesu na ramki) oraz wykonywania tłumaczenia adresów logicznych (strona + przesunięcie na stronie). Także sama tablica stron procesu procesu może być stronicowania (mamy tablicę która informuje nas że przypisania stron w danym zakresie adresów są przechowywane w jakiejś ramce).

Strony i ramki mogą być współdzielone pomiędzy procesami (np. przy rozgałęzianiu procesu strony są kopiowane dopiero gdy zajdzie taka potrzeba). W przypadku braku miejsca w pamięci fizycznej wybrane strony nieaktywnego aktualnie procesu mogą być umieszczane na dysku (swap[570]). Niekiedy może to powodować szamotanie procesu[571] polegające na zbyt dużej liczbie wymian stron. Zawsze jednak prowadzi to do konieczności ustalania które strony najlepiej jest przenieść na dysk. Optymalne byłoby przenoszenie tych które najdłużej nie będą potrzebne (jednak z oczywistych względów jet to praktycznie nie do zrealizowania). Stosuje się różne algorytmy tego wyboru:

- FIFO - usuwamy najdłużej będącą w pamięci
- LRU - usuwanie tej do której najdawniej się odwoływano (licznik czasu, bit odniesienia, bit odniesienia w określonym czasie, bit modyfikacji)
- LFU - usuwamy z najmniejszą liczbą odwołań
- MFU - usuwamy z dużą liczbą odwołań

Alternatywną (i mniej wymagającą) wobec stronicowanie metodą zarządzania pamięcią jest segmentacja[572]. W przypadku architektury x86 jest ona zawsze wykorzystywana jednak może być przykryta dużym segmentem na którym wykorzystujemy stronicowaniem. Wspomniany już mechanizm pamięci wirtualnej[573] często rozumiany jest tylko jako stronicowaniem (lub segmentacja) na żądanie. Polega to na tym iż strony mapowane są na ramki dopiero w momencie zapisu do segmentu pamięci należącego do

Zobacz w Sieci: Kurs Pisania OS[574], Intel 8259[575], Input/Output Base Address[576].

Systemy POSIXowe

Bash

Bash[577] jest jednym z najpopularniejszych interpretatorów poleceń (jest on oparty na składni sh[578]), podobnie jak inne umożliwia on zarówno pracę interaktywną (wydawanie kolejnych poleceń przez operatora systemu), jak i pracę w trybie wsadowym (czyli wykonywanie skryptów powłokowych, stanowiących zapisane w pliku tekstowym ciągi poleceń - oczym więcej w części poświećonej programowaniu w bash).

Praca interaktywna polega na wprowadzaniu ciągu poleceń (przy pomocy jakiejś formy klawiatury), które są wykonywane i ich wyniki widoczne są w postaci zmian w plikach/systemie plików lub w postaci komunikatów wyświetlanych na ekranie. Normalnie odpalony `bash` uruchamia się w trybie pracy interaktywnej, uprzednio czytając swoje pliki konfiguracyjne (m.in. `~/.bashrc` - prosty przykład, bardziej rozbudowany plik zamieszczam w konfiguracja środowiska pracy). W trybie tym możemy:

- korzystać z historii linii poleceń - przewijanie przy pomocy strzałek w górę i dół, wyszukiwanie przy pomocy Ctrl+R, z kolei umieszczając w pliku konfiguracyjnym `export HISTIGNORE="&:ls:[bf]g:cd:exit:kill *"` możemy ustawić ignorowanie powtórzeń (&) oraz wybranych komend (w podanym przykładzie kill z dowolnymi argumentami, ale nie killall);
- korzystać z autouzupełniania wpisywanych nazw poleceń / ścieżek przy pomocy Tab, aby rozbudować tą funkcjonalność warto zainteresować się "bash completion" i dodać do swojego pliku konfiguracyjnego `. /etc/bash_completion`;
- korzystać z zdeklarowanych w pliku konfiguracyjnym aliasów dla często wydawanych poleceń z zestawem opcji np. `alias 'wlacz_komputer'='/usr/sbin/etherwake MAC:ADRES:KOMPUTERA:KTORY:LUBIMY:WLACZAC'` lub `function kman { command konqueror man:/"$@"; }` (command zapewnia że wywołamy komendę a nie funkcję basha lub alias);
- edytować linię polecenia (poruszanie się po niej strzałkami przód tył i home/end, kasowanie przy pomocy backspace/delete oraz wprowadzanie zmian).;
- wykorzystywać wszystkie polecenia używane zazwyczaj w trybie wsadowym.

BASHRC

```
# konfiguracja ogólna

mesg y

umask 0022

export TEXINPUTS=.:~/.tex:
export PATH=$HOME/.bin:"${PATH}"
export PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '

alias 'cp'="cp -i"
alias 'mv'="mv -i"
alias 'rm'="rm -i"

alias 'ls'="ls --color=auto --time-style=+\" %Y-%m-%d %H:%M:%S \" -h"
alias 'date_iso'='date +"%A,  %Y-%m-%d %H:%M:%S %z (%Z)%n time stamp: %s"'
alias 'date_iso_utc'='date_iso --utc'
alias 'bc'='bc <(echo scale=3)'

function pakiety-bck() { mkdir -p /etc/pakiety-bck/; tar -czf /etc/pakiety-bck/`date '+
%FT%T'`.tgz /var/lib/dpkg/status /var/lib/dpkg/statoverride /var/lib/dpkg/diversions 
/var/lib/dpkg/alternatives /var/lib/dpkg/info /var/lib/apt/extended_states 
/var/lib/aptitude/pkgstates; }
```

```
function arecord() {
 if [ $# = 0 ]; then
  arecord -fcd `date +"%F_%T.wav"`;
 else
  /usr/bin/arecord $@
fi
}

alias y='youtube-dl -b -t'
alias bmv='bzr mv'

function y2p() {
 filename=`echo "$1" | awk 'BEGIN{RS="[&?]"; FS="="} $1=="v"{print $2}'`
youtube-dl -o /tmp/$filename.flv $1 &
 sleep 2
 kmplayer /tmp/$filename.flv
kill %1
}

. /etc/bash_completion
```

Standardowe wejście wyjście i przekierowania

Do przekierowywania wejść i wyjść poleceń służą znaki:
| - przekierowanie wyjścia (polecenia z lewej strony) do programu (polecenia po prawej stronie)
> - przekierowanie wyjścia do pliku
< - pobieranie wejścia z pliku.

możliwe jest też przekierowanie stdout dwóch poleceń jako wejść plikowych - np. `diff <(cat plik1) <(cat plik2)`. Dzięki konstrukcji `exec 1> plik.txt` można przekierować całe standardowe wyjście do pliku, natomiast ``zmienna=`cat` `` umożliwi wczystanie standardowego wejścia do pliku. Warto w tym miejscu wspomnieć także o programie `tee`, który wypisuje swoje standardowe wejście do pliku i na standardowe wyjście jednocześnie.

Łączenie poleceń

Do łączenia poleceń oprócz przekierowywania wejść i wyjść służą także inne znaki:

- rozdzielane średnikiem (lub niecytowanym znakiem nowej linii)- wykonywane wszystkie po kolei
- rozdzielane || (LUB logiczne) wykonywane są aż któreś zwróci 1
- rozdzielane && (I logiczne) wykonywane są od lewej aż któreś zwróci 0
- rozdzielane & - polecenie poprzedzające przechodzi w tło i zwraca zero polecenie następujące wykonywane jest normalnie

Polecenia możemy także grupować przy pomocy nawiasów klamrowych `{ polecenia; }` (uwaga średnik i spacje, które możemy ewentualnie zastąpić znakami nowej linii są konieczne) lub nawiasów okrągłych `(polecenia)`, wtedy wykonywane są w podpowłoce (uruchomiony jest osobny proces basha).

Sterowanie procesami

W systemach unixowatych z uruchomionymi procesami możemy komunikować się m.in. przy pomocy sygnałów. Do ich wysyłania służy program `kill`. Warto jednak wspomnieć także o kombinacji klawiszy [Ctrl]+[c] wysyłającej sygnał SIGINT (nr. 2, przerwanie) oraz [Ctrl]+[/] wysyłającej sygnał SIGQUIT (nr. 3, zakończenie programu). Jeżeli jest już mowa o kombinacjach specjalnych kończących działanie programu to trzeba wspomnieć też

o [Ctrl]+[d] wysyłającej znak ASCII[579] o numerze 4 czyli EOT (koniec transferu) czyli informację o końcu wprowadzania danych tekstowych (nie jest to EOF ani ETX mające podobne znaczenie). Kombinacja ta wprowadzana np. w bashu kończy jego działanie.

Działanie w tle

Celem uruchomienia polecenia w tle podajemy na końcu linii poleceń znak &. Listę tak uruchomionych zadań z aktualnego shela możemy wyświetlić poprzez polecenie `jobs` a następnie przenieść na pierwszy plan poleceniem `fg %n` lub zakończyć poprzez `kill %n` (gdzie n jest wyświetlonym przez jobs numerem zadania). Warto też wspomnieć o kombinacji [Ctrl]+[z] (wysyłającej sygnał SIGTSTP) zawieszającej wykonywanie aktualnego programu i powodującej ponowne przejście do shela, następnie można spowodować odwieszenie i przesunięcie w tło tak przerwanego procesu poprzez `bg [%n]`. Procesy uruchomione w tle możemy także odłączyć od powłoki z wykorzystaniem `disown`.

Sztuczki z *[]$()?

Warto też wspomnieć o możliwości korzystania ze znaków wieloznacznych: * - dowolny ciąg znaków niezaczynający się od kropki, ? - dowolny jeden znak, [-xawA-F] - dowolny ze znaków wymienionych : -, x, a, w oraz przedział od A do F. Aby podejrzeć jakie pliki zostaną dopasowane do naszego wzorca polecam `echo` - np. `echo ?[a-z]*` wypisze wszystkie pliki mające na drugiej pozycji mała literę (w Unixach wielkość liter ma znaczenie).

Jest także możliwość przekazania wyniku polecenia jako parametru innego polecenia: `inne_polecenie `polecenie``, jeżeli chcemy jeszcze bardziej zagnieżdzać zamiast ` w Bashu możemy używać konstrukcji z `$()`, np. `inne_polecenie $(polecenie)`.

Przy użyciu podwójnych nawiasów możemy korzystać z wbudowanej prostej arytmetyki stałopozycyjnej - `echo $((2+2))` wypisze 4. Podobnie działają też pojedyncze nawiasy kwadratowe - np. `echo $[2+2]` wypisze 4, należy jednak pamiętać iż metoda z nawiasami kwadratowymi (w odróżnieniu od okrągłych) jest nieprzenośna na inne sh. Prostą artmetykę można realizować także przy pomocy polecenia let - np. `let zm++` zwiększy wartość zm o jeden, jednak jest to równierz metoda nieprzenośna na inne sh. Do bardziej zaawansowanej matematyki wykorzystać można program bc - `echo "2+2" | bc` robi to samo … (więcej `man 1 bc`).

Kolejną sztuczką jest wbudowane przetwarzanie napisów przy pomocy `{%}`, np. `echo ${PATH%:*}` wybisze zawartość zmiennej $PATH bez ostatniego członu (po ostatnim :), proktycznym zastosowaniem może być np. masowa zamiana rozszerzeń plików `for f in *.txt ; do mv "$f" "${f%.txt}.htm"; done`. Inne zastosowanie klamerek (podstawienia wielkrotne każdy z każdym) pokazuje `echo A{7,8}X{1,2}`. Warto także wspomnieć o różnicy między `echo $zmienna` a `echo "$zmienna"`, gdy zmienna zawiera znaki nowej linii (w pierwszym wypadku zostaną ope zastąpione spacjami). Więcej o tego typu sztuczkach w części o zmiennych.

Podstawowe polecenia Unix'a

Uzyskiwanie pomocy

Poniżej najważniejsze komendy przeznaczone do przeglądania pomocy wbudowanej w system. Należy także wspomnieć iż w wielu komendach możemy także uzyskać pomoc za pomocą opcji `--help` lub `-h`. Ponadto dla wielu programów/bibliotek itp dostępna jest dokumentacja w `/usr/share/doc`.

- `man [nazwa polecenia]` - podręcznik elektroniczny, warto również zobaczyć reakcję polecenia na parametr `--help`
- `info [nazwa polecenia]` - nowa wersja podręcznika systemowego
- `pinfo` - wygodna przeglądarka dokumentacji info w trybie tekstowym
  Homepage: http://pinfo.alioth.debian.org/
- `apropos [tekst]` - listuje strony podręcznika związane z danym tekstem, `apropos -s 1 ''` | less pozwala zapoznać się z spisem wszystkich stron w rozdziale pierwszym (o tym co jest w którym rozdziale mozna zobaczyć w `man man`)

Uzyskiwanie informacji o plikach, katalogach, poruszanie się w strukturze drzewa katalogów

- `cd [ścieżka]` - zmiana bieżącego katalogu

  warto zauważyć iż katalog bieżący oznaczamy `.`, nadrzędny oznaczamy `..`, ścieżki zaczynające się od ukośnika / oznaczają ścieżki bezwzględne (od korzenia systemu plików), natomiast katalog domowy oznacza się ~
- `ls [opcje] [ścieżka]` - listowanie zawartości katalogu

  do ważniejszych opcja należy zaliczyć `-a` powodująca wyświetlanie plików ukrytych (zaczynających się od kropki) i `-l` wyświetlająca pliki w formie listy z szczegółowymi informacjami (uprawnienia, rozmiar, właściciel, grupa, rozmiar)
- `tree` - listowanie drzewa katalogów
- `du [opcje] [ścieżka]` - wyświetla informacje o miejscu zajmowanym przez katalogi/pliki

  do ważniejszych opcji należy zaliczyć `-k` - podaje ilość zajmowanego miejsca w kilobajtach, `-s` - podaje tylko sumę dla każdego katalogu
- `pwd` - wypisuje ścieżkę bieżącego katalogu
- `mount` - montuje systemy plików

  warto zwrócić szczególną uwagę na opcję montowania bind (podawaną po `-o` lub jako `--bind`) umożliwiającą montowanie katalogu w innym katalogu oraz na `-o remount` umożliwiającą zmianę parametrów podmontowanego systemu plików
- `umount` - odmontowywuje systemy plików
- `fdisk` - `cfdisk`, `sfdisk` - modyfikowanie / oglądanie podziału dysku na partycje
- `realpath` - narzędzie zamieniające ścieżki utworzone w oparciu o linki symboliczne na ściezki rzeczywiste (równoważne pod względem funkcjonalnym `/bin/readlink -f`)
  Homepage: http://packages.debian.org/squeeze/realpath
- `readlink` - narzędzie do pokazywania wartości linków symbolicznych i rozwijania ścieżek, np. jednolinijkowiec `for l in *; do if [ -L "$l" ]; then f=`readlink "$l"`; \rm "$l"; cp "$f" "$l"; fi; done` pozwala na zamianę linków na rzeczywiste pliki w bierzącym katalogu

Zmiany w strukturze katalogów, kopiowanie przenoszenie usuwanie

- `mkdir [ścieżka]` - tworzenie katalogów
- `rmdir [ścieżka]` - usuwanie katalogów
- `cp [opcje] [źródło] [ce]` - kopiowanie plików

  do ważniejszych opcji należy zaliczyć `-r` kopiuje podkatalogi wraz z plikami, `-f` pliki w katalogu docelowym posiadające identyczne nazwy będą nadpisywane bez pytania, `-i` oczekuje potwierdzenia przed nadpisaniem, `-v` wyświetla nazwę każdego pliku podczas kopiowania
- `ln [opcje] [źródło] [cel]` - służy do tworzenia **dowiązań twardych** oraz symbolicznych, domyślnie tworzy dowiązanie twarde - drugi równoprawny wpis o pliku (w szczególności plik zostanie skasowany, gdy wszystkie dowiązania twarde do niego i on sam zostaną usunięte)

  do ważniejszych opcji należy zaliczyć `-s` - tworzy **dowiązanie symboliczne** (plik wskazujący na położenie celu, konieczne przy dowiązaniach do katalogów, a także plików położonych na innych partycjach), `-f` - pliki w katalogu docelowym posiadające identyczne nazwy będą nadpisywane bez pytania
- `mv [opcje] [źródło] [cel]` - przenoszenie/zmiana nazwy pliku

  do ważniejszych opcji należy zaliczyć `-f` pliki w katalogu docelowym posiadające identyczne nazwy będą nadpisywane bez pytania, `-i` oczekuje potwierdzenia przed nadpisaniem, `-v` wyświetla nazwę każdego pliku podczas kopiowania
- `rm [opcje] [ścieżka]` - usuwanie plików lub katalogów, jeżeli brak jest odpowiednich praw do zapisu, wyświetlana jest prośba o potwierdzenie.

do ważniejszych opcji należy zaliczyć `-f` brak pytań przy każdym usuwanym pliku, `-i` oczekuje na potwierdzenie, `-r` podkatalogi i pliki również zostaną usunięte, `-v` wypisuje nazwę pliku przed usunięciem

- `symlinks` - narzędzie do weryfikacji, upraszczania, zmiany na względne, itp linków symbolicznych
- `rdiff-backup` - i `pdumpfs` - różnicowe kopie zapasowe ...
- `rsync` - program do synchronizacji plików i drzew katalogów przez sieć (różnicowe kopie zapasowe), bardzo wygodne jesty użycie poprzez ssh i dynamicznie urychamiany rsynk po drugiej stronie przy użyciu opcji `-e 'ssh'`
  Homepage: http://rsync.samba.org
- `dd` - tworzenie kopi urządzenia / pliku obrazu
- `ddrescue` - dd z lepszą tolerancją błędów odczytu
  Homepage: http://www.gnu.org/software/ddrescue/ddrescue.html

Pakowanie i kompresja

- `tar` - programy do pakowania/rozpakowywania
- `gzip` - kompresuje podany plik (nie łączy wielu plików w jedno archiwum)
- `bzip2` - kolejny kompresor
- `split` - dzieli plik na części (łączyć można np. cat'em)
- `pax` - rozbudowany archiwizer umożliwiający również kopiowanie drzewa katalogów
- `cpio` - również archiwizery ...
- `afio` - również archiwizery ...
- `compress` - popularne narzędzie kompresji plikó
  Homepage: http://ncompress.sourceforge.net/
  Debian: ncompress[580]
- `zoo` - narzedzie do kompresji i dekompresji
  Homepage: http://www.ibiblio.org/pub/packages/ccic/software/unix/utils/zoo210.tar.gz
- `zip` - program do obsługi archiwów ZIP
  Debian: zip[581]
- `unrar` - rozpakowywacz archiwów RAR
  Debian: unrar-free[582]
- `unzip` - rozpakowywacz archiwów RAR
  Debian: unzip[583]

Atrybuty i uprawnienia

- `touch` - zmienia daty modyfikacji, dostępu, ... do pliku, pozwala na łatwe tworzenie pustych plików
- `chown [opcje] [właściciel] [ścieżka]` - zmiana właściciela pliku

  do ważniejszych opcji należy zaliczyć `-R` powodująca zmienianie również w podkatalogach
- `chgrp [opcje] [grupa] [ścieżka]` - zmiana grupy pliku

  do ważniejszych opcji należy zaliczyć `-R` powodująca zmienianie również w podkatalogach
- `chmod [opcje] [uprawnienia] [pliki]` - zmiana prawa dostępu do pliku(ów)

  do ważniejszych opcji należy zaliczyć `-R` powodująca zmienianie również w podkatalogach
  Unixowe uprawnienia do plików składają się z trzech członów: uprawnienia dla właściciela (u), grupy (g) i pozostałych użytkowników (o). W każdym z członów mogą być przyznane uprawnienia do czytania (r), pisania (w) i wykonywania (x); w odniesieniu do plików jest to intuicyjne (uprawnienie do wykonywania jest potrzebne do uruchomienia programów), natomiast w stosunku do katalogów wygląda to następująco: uprawnienia do czytania pozwalają na wylistowanie zawartości, do wykonania pozwalają na dostęp, do zawartości katalogu (wejścia do niego) do pisania na tworzenie nowych obiektów wewnątrz niego i zmienianie nazw istniejących. Warto zaznaczyć iż program w pliku nie mającym praw wykonywlności też moży być wykonany (w przypadku skryptu poprzez uruchomienie interpretatora i podanie tego programu jako odpowiedniego argumentu wywołania, w przypadku typowych binarek poprzez wywołanie `/lib/ld-2.3.6.so` z tym plikiem i jego opcjami jako paramtrem.
- `chattr` - zmienia atrybuty plików związane z systemem plików
- `getfacl` - odczyt uprawnień związanych z listami kontroli dostępu

- `setfacl` - ustawianie uprawnień związanych z listami kontroli dostępu.

  ACL jest opcjonalnym mechanizmem który (na wspierających go systemach plików) pozwala na definiowanie uprawnień do pliku dla dodatkowych użytkowników i grup - plik ma nadal swojego właściciela, grupę i wszystkich pozostałych, ale przed prawami dla "others" wchodzą prawa użytkowników i grup definiowanych w ACL (w najpopularniejszej minimalnej wersji według schematu rwx). Wypadkowe prawa obliczane są jako suma wynikła z praw użytkownika i grup do których należy.

  ACL pozwala ponadto definiować uprawnienia domyślne dla nowo powstałych plików w katalogu (są one opcją katalogu).

  Więcej szczegółów o ACL - Access Control Lists w Linuxie[584], Mechanizmy lokalnej kontroli dostępu (ACL)[585])

Wyszukiwanie plików i w plikach, porównywanie plików, ...

- `find [opcje]` - przeszukiwanie systemu plików

  do ważniejszych opcji należy zaliczyć `-name nazwa-pliku` wyszukanie pliku nazwa-pliku, `-size +N` wyszukanie pliku o rozmiarze większym niż N
- `locate` - wyszukuje pliki po nazwie korzystając z specjalnej bazy danych (wyniki mogą być mniej aktualne niż przy find, za to szybsze)
- `grep [opcje] [ścieżka]` - wypisywanie linijek zawierających słowa kluczowe (oraz po skorzystaniu z odpowiednich opcji linijek sąsiadujących albo linii nie pasujących), warto tu zwrócić uwagę na możliwość podania kilku wzorców (łączone LUB) lub nawet całego pliku wzorców (można w ten sposób porównywać pliki)
- `diff` - porównuje pliki, do uzyskania pliku zmodyfikowanego z orginalnego i łaty wygenerowanej przez `diff` służy program `patch`
- `comm` - wyświetlanie linii występujących tylko w jednym (wskazanym) z dwóch posortowanych plików lub linii wspólnych
- patchutils[586] - zestaw narzedzi do pracy z patch'ami
- quilt[587] - narzedzie do pracy z seriami patch'y
- `vbindiff` (Visual Binary Diff) - porównuje wizualnie pliki binarne (interfejs ncurces)
  Homepage: `http://home.comcast.net/~chris-madsen/vbindiff/`
- `patch` - stosuje na pliku łatę zwracaną przez diff
- `sort` - sortuje ...
- `wc` - liczy linie, wyrazy, znaki, ...
- `file [plik]` - rozpoznaje typ pliku

Wyświetlanie i edycja plików

- `cat` - wypisywanie i łączenie plików
- `more` - przeglądanie tekstu ekran po ekranie
- `less` - przeglądarka tekstu ekran po ekranie o większych możliwościach od `more`(nie musi czytać całego dokumentu, coznacznie szybsze przy dużych dokumentach oraz posiada możliwość przeglądanie dokumentu w tył)
- `head` - wypisuje początek pliku
- `tail` - wypisuje koniec pliku
- `multitail` - tail lepiej operujący na wielu plikach równocześnie
- `tac` - wypisuje plik od konca (ostatnia linia jako pierwsza
- `ed` - prosty "jednolinijkowy" edytor tekstowy
- `vim` - legendarny edytor tekstowe

  podstawowe komendy VIMa: `i` - tryb wstawiania, `Insert` - zmiana trybu wstawiania i zastępowania, `Esc` - wyjęscie z trybu wstawiania/zastępowania, `y` - kopiuje, `d` - kasuje (po `y`, `d` mozna podac np `20l` lub `20{strzalka w prawo}` co oznacza 20 kolejnych znakow, `w` oznacza slowo, podwojenie `y` lub `d` oznacza kopiowanie/kasowanie całej linii), `x` - kasuje tyle znakow jaka liczba poprzedzony, `p` - wkleja po, `P` - wkleja przed, `/` - szukanie, `N` - poprzednie znaleznien, `n` - nastepne, `1,$s#szukany#zamiennik#g` - zastępowanie w całym pliku (bez adresu `1,$` - w bierzącej linii), `:w` - zapis `:q` - wyjście `:q!` - wujście bez zapisywania `:wq` - zapis i wyjście `:next` -

następny plik `:previous` - poprzedni plik `:split` - podział okna `Ctrl+W` - przełączanie między oknami
Ściąga do vi (pl)[588], Ściąga do vi (en)[589] - kilkustronicowe broszurki z najważniejszymi poleceniami, graficzny tutorial do vim'a[590] (i alternatywna wersja[591]), Praktyczny Vim[592]
Warto także zainteresować się sprawdzaniem pisowni w vim: instalujemy paczkę vim-spellfiles-pl (jest w experimental, instaluje się w `/usr/share/vim/vim70/spell/`), w `~/.vimrc` wpisuje: `set spelllang=pl,en`, a następnie sprawdzamy przy pomocy `:set spell`.
Homepage: `http://www.vim.org/`

- `emacs` - inny kultowy edytor tekstowy
  Ściąga do Emacsa[593] - kilkustronicowa broszurka z najważniejszymi poleceniami
  Homepage: `http://www.gnu.org/software/emacs/emacs.html`
- `joe` - kolejny edytor tekstowy
- `sed` - edytor strumieniowy, umożliwia np. podmianę pierwszego wystąpienia napisu w pliku `sed -e '0,/zastepowany/ s#zastepowany#zastepujacy#'` lub wszystkich wystąpień - `sed -e 's#zastepowany#zastepujacy#g'`
- `xmlstarlet` - comand line'owe narzędzie do przetwarzania, modywfikowania, … plików xml (można powiedzieć taki odpowiednik sed/awk dla xml'a)
- `cut` - wybiera z pliku zadany zestaw kolumn
- `paste` - łączenie plików (wg. zadanych kryteriów) poprzez łączenie linii
- `join` - łączenie plików (wg. zadanych kryteriów) poprzez wstawianie linii
- `rev` - wypisuje plik odwracając kolejność znaków w każdym wierszu
- `nl` - wypisuje plik z numerowanymi liniami
- `pr` - konwersja plików tekstowych do drukowania (np. dodanie marginesu)
- `tr` - zamienia jeden zestaw znaków innym (niestety ma problem z znakami dwubajtowymi)
- `konwert` - oparty na perlu program do konwersji różnych kodowań, posiadający predefiniowane zestawy (np. `konwert utf8-ascii` usunie znaki narodowe itp z pliku kodowanego w utf-8, natomiast filtr qp-8bit pozwoli zamienić kodowanie quoted printable na normalne 8 bitowe, a rtf-8bit zrobi to z kodowaniem rtf'u)
  Homepage: `http://qrczak.ids.net.pl/programy/linux/konwert/`
- `str` - bardziej inteligentny od `tr` program wchodzący w skład pakietu konwert służy do zamieniania napisów z jednego zestawu napisami z drugiego (w odróżnieniu od tr nadaje się do znaków kilku bajtowych - utf8, lepiej od sed'a sprawdza się przy zmianie kodowań - patrz przykład w manualu)
- `iconv` - zmienia kodowanie znaków w pliku
- `mewencode` - program (stanowiący część pakieu narzędzi dodatkowych dla kilenta pocztowego Mew) do obsługi kodowań mime (w tym Quoted-Printable, base64), m.in. zmienia kodowanie base64 na 8 bitowe
  Homepage: `http://www.Mew.org/`
  Debian: mew-bin[594]
- `mimedecode` - narzędzie do obsługi kodowania MIME
  Homepage: `http://www.freesoft.org/CIE/FAQ/mimedeco.c`
- `qprint` - program do kodowania i dekodowania "Quoted-Printable"
  Homepage: `http://www.fourmilab.ch/webtools/qprint/`
- `base64` - program do kodowania i dekodowania base64
  Homepage: `http://www.fourmilab.ch/webtools/base64/`
- `hexcat` - wypisywanie plików jako liczby szesnastkowe
- `od` - wypisywanie plików jako liczby oktalne, szesnastkowe, …
- `xxd` - wypisywanie plików jako liczby szesnastkowe
- `hexdump` - wypisuje zawartość pliku interpretując go jako plik binarny, np. `hexdump -e '1/4 "%08d " 1/2 "%05d " 2/1 "%02x \n"' plik` każde kolejne 8 bajtów będzie interpretował jako 32 bitowa liczba całkowita (4 bajty), 16 bitowowa liczba całkowita (2 bajty) i dwie liczby jedno bajtowe (wyświetlane szesnastkowo).
- `bvi` - `hexer` - edytor binarny o interfejsie przypominającym vi
- `hexcurse` - edytor binarny o interfejsie opartym o ncurses
- `indent` - automatyczne formatowanie kodu C
- `tidy` - automatyczne formatowanie xml/html/…
- `expand` - konwersja tabulacji na spacje
- `unexpand` - konwersja spacji na tabulacje

Planowanie zadań

- `crontab` - pozwala oglądać i edytować tablice zaplanowanych zadań (dla cron'a). Warto tu też wspomnieć o programie "anacron" będącym mniej przestrzegającą reguł czasowych odmianą cron'a
- `at` - pozwala jednorazowo zaplanować zadanie

Procesy, ich parametry i informacje o nich

- nice - uruchamia program z zmodyfikowanymi priorytetem szeregowania (im wyższy tym program jest bardziej dyskryminowany przy przełączaniu zadań)
- renice - zmienia priorytet szeregowania procesu
- ionice - uruchamia program z zmodyfikowanymi priorytetem IO lub zmienia te priorytety dla wskazanego procesu
- cpulimit - umożliwia ograniczenie wykorzystania procesora przez program
  Homepage: http://cpulimit.sourceforge.net/
- kill - przesyła sygnał do procesów o podanych PID
- killall - przesyła sygnał do procesów o pasującej nazwie
- ps - wyświetla aktualnie działające procesy i informacje o nich (dla ps polecam opcje -Alf ...)
- pgrep - wyświetla filtrowaną listę aktualnie działających procesów
- pstree - wyświetla drzewo aktualnie działających procesów
- pwdx - podaje katalog roboczy podanego procesu
- top - monitorowanie procesów obciążających CPU, pamięć, ...
- iotop - monitorowanie procesów obciążających I/O
- fuser - informacje o procesach korzystających z plików ...
- lsof - informacje o plikach otwartych przez program (patrz też /proc/PID/fd/)
- socklist - wyświetla listę otwartych socketów
- vmstat - informacje o obciążeniu systemu
- ipcs - informacje o wykożystaniu pamięci współdzielonej

Informacje o użytkownikach, systemie, ...

- id - informacja o aktualnym użytkowniku
- whoami - informacja o aktualnym użytkowniku
- w - kto obecnie zalogowany
- who - kto obecnie zalogowany (ale np. z opcją -r aktualny poziom działania systemu i czas uruchomienia)
- finger [nazwa uzytkownika] - informacja o użytkowniku
- last - ostatnio logujący się użytkownicy
- lastb - ostatnie nieudane logowania
- lastlog - ostatnie logowania użytkowników
- uname - informacja o systemie
- lsb_release - informacja o systemie zgodnym z LSB
- printenv - zmienne środowiskowe wypisywanie oraz przekazywanie do uruchamianego programu
- env - zmienne środowiskowe wypisywanie oraz przekazywanie do uruchamianego programu
- setenv - zdefiniowanie zmiennych środowiskowych
- unsetenv - skasowanie zmiennych środowiskowych
- free - informacja o pamięci
- uptime - informacja o tym jak długo działa system ...
- getent - informacje z baz administracyjnych
- time - czas działania programu
- dumpkey - zwraca kody wciskanych klawiszy ...
- xconsole - pokazuje komunikaty wyrzucane na konsolę w środowisku graficznym (wymaga dostępu do /dev/xconsole, prawa dostępu do tego pliku są ustalane w /etc/init.d/sysklogd

Komunikacja z i pomiędzy użytkownikami

- write [nazwa użytkownika] [terminal] (nwrite) - wysyła komunikat do danego użytkownika

  Warto tutaj wspomnieć o możliwości kierowania komunikatu na wybrany terminal (m.in. /dev/tty* w "przypadku" prawdziwych terminali, /dev/pts/* w przypadku X-serwera, xterminali i sesji ssh, zerowy to najczęściej X-serwer). W przypadku posiadania stosownych uprawnień można też pisać bezpośrednio do tych plików.
  Homepage: ftp://ftp.csua.berkeley.edu/pub/nwrite/
  Debian: nwrite[595]
- wall - wiadomość rozgłoszeniowa (do wszystkich zalogowanych użytkowników)
- talk [nazwa użytkownika] - rozmowa z użytkownikiem

— `mesg` - pozwala zezwolić bądź zabronić na wysyłanie wiadomości przez write na twój terminal ...

Różne użyteczne narzędzia ...

— `cal` - kalendarz
— `date` - data i czas, program ten potrafi także przeliczać datę i czas - np. `date -d @847103830 '+%Y-%m-%d %H:%M:%S'`, `date -d '1996-11-04 11:37:10' '+%s'`, `date -d '1996-11-04 11:37:10 +3week -2days'`
— `bc` - kalkulator (warto rozważyć utworzenie sobie aliasu alias `'bc'='bc <(echo scale=3)'`, aby zaraz po starcie mieć 3 miejsca po przecinku)
— `calc` - (znany także jako apcalc) zaawansowany kalkulator o składni podobnej do C umożliwiający wykonywanie mi.in. operacji logicznych (w tym binarnych)
— `numconv` - program do konwersji systemów liczbowych oprócz systemów pozycyjnych o różnych podstawach obsługuje także inne systemy liczbowe - np. rzymski
— `ipcalc` - kalkulator IP (pozwalający na obliczanie adresów sieci rozgłoszeniowych, zmianę notacji itd)
— `sipcalc` - kalkulator IP (pozwalający na obliczanie adresów sieci rozgłoszeniowych, zmianę notacji itd)
— `seq` - wypisuje kolejne liczby (przydane np. w for)
— `xargs` - pobierz argumenty z stdin
— `strings` - wypisuje sekwencje znaków drukowanych (określanie zawartości plików nietekstowych)
— `clear` - czyści terminal
— `reset` - przywraca ustawienia terminala (np. gdy po wyświetleniu pliku binarnego są dziwne krzaczki)
— `script` - tworzy zapis sesji (stdin, stdout, ...)
— `which [tekst]` - zwraca wykonywaną ścieżkę / polecenie przy wykonywaniu podanego tekstu
— `echo` - druk na standardowe wyjście
— `whois` - informacje o domenie
— `host` - informacje DNS'owe
— `mkfifo` - tworzy łącze nazwa (specjalny plik który może być wykorzystywany do komunikacji między procesami na zasadzie podobnej do |)
— `gnuplot` - rysowanie wykresów; więcej o gnuplocie
— `xinit` - uruchamia program w srodowisku graficznym (wraz z startem srodowiska), np. `xinit -e __WYBRANY_PROGRAM__ -- :1` uruchomi X'y na ekranie 1 (zazwyczaj Alt+Ctrl+F8) a w nich wskazany program
— `chvt` - komenda do przełączania aktywnego terminala wirtualnego (w tym terminali na których pracuje Xserwer),komenda jest odpowiednikiem kombinacji ALT+CTRL+F*
— `openvt` - wykonanie komendy na zadanym terminalu wirtualnym (nie mylić z emulatorem terminala - np. xterm)
— `mt` - obsługa napędów taśmowych (streamerów), np. `mt -f /dev/rmt/0 rewind` przewinięcie do początku taśmy w wskazanym czytniku ...

Narzędzia diagnostyczne i administracyjne

Oprócz opisanych poniżej narzędzi istnieje wiele innych programów związanych z tą tematyką (zobacz w innych artykułach na temat linuxa na stronie). Wartościowe są również informacje zawarte w systemie `/proc`, `/sys` oraz w logach (`/var/log`).

— `su` - przełącza użytkownika (podobnie działa stworzony dkla kde `kdesu`, w przypadku `su` aby przełączony używtkownik miał dostęp do "naszego" x serwera wcześniej wydajemy `xhost LOCAL:nazwa_uzytkownika_na_ktorego_przelaczamy`)
— `sudo` - program pozwalający na wykonywanie uprzywilejowanych komend przez wyznaczonych użytkowników Homepage: `http://www.sudo.ws/`
— fuse-utils[596] - narzędzia do korzystania z FUSE[597]
— fakechroot[598] - nardzedzie do uruchamiania programu z podmienionym katalogiem głównym
— `stty` - konfiguruje terminal (np. port szeregowy /dev/tyyS* , później możemy korzystać z niego np. za pomocą cat, echo, ... i przekierowań potoków - np. `cat /dev/ttyS0 > ~/serial.log` będzie logowało informacje z portu do wskazanego pliku)
— `xset` - konfigurowanie ekranu x-serwera, np. `sleep 1; xset dpms force off` spowoduje wyłączenie (uśpienie) monitora
— `adduser` - dodawanie użytkownika
— `addgroup` - dodawanie grupy
— `deluser` - usuwanie użytkownika

— `delgroup` - usuwanie grupy
— `passwd` - zmiana hasła
— `chage` - informacja o ważności hasła
— `shutdown` - wyłączanie i restartowanie systemu (komputera), podobnie do operowania na poziomach `init`
— `dmesg` - logi jądra
— `lspci` - listowanie tego co siedzi na pci, pcmcia, … (aktualizacjia bazy identyfikacyjnej `update-pciids`)
  Debian: pciutils[599]
— `lspcmcia` - listowanie slotów PCMCIA
  Debian: pcmciautils[600]
— `pcardctr` - kontrola PCMCIA - wyłączanie, włączanie urządzeń itp
  Debian: pcmciautils[601]
— `lsusb` - listowanie tego co siedzi na usb (aktualizacjia bazy identyfikacyjnej `update-usbids`)
  Debian: usbutils[602]
— `lsscsi` - listowanie tego co siedzi na SCSI (oraz interfejsach tak widzianych - np. SATA, FC, …)
— `dmidecode` - odczyt informacji o sprzęcie
— `ddcprobe` - odczyt informacji o monitorze
— `lshal` - odczyt informacji na temat HAL
— `ldconfig` - konfiguracja bibliotek dynamicznych (m.in. określanie ścieżek w jakich są wyszukiwane), patrz też `/etc/ld.so.conf`
— `ldd` - wyświetlanie wykorzystywanych przez program bibliotek dynamicznych
— `hddtemp` - nardzędzie do kontroli temperatury dysku twardego
  Homepage: `http://www.guzu.net/linux/hddtemp.php`
— `smartctl` - nardzędzie do odczytu informacji S.M.A.R.T.[603] dotyczących dysku twardego
  Homepage: `http://smartmontools.sourceforge.net/`
  Debian: smartmontools[604]
— `hdparm` - ustawienia i pomiary prędkości dysków twardych (np. opcja -S ustawia czas po jakim usypiany jest dysk)
  Homepage: `http://www.ibiblio.org/pub/Linux/system/hardware`
— `sdparm` - narzędzie uzupełniające hdparm w stosunku co urządzeń podsystemu SCSI
  Homepage: `http://sg.danny.cz/sg/sdparm.html`
— `mbmon` (xmbmon) - monitorowanie parametrów pracy płyty głównej
  Homepage: `http://www.nt.phys.kyushu-u.ac.jp/shimizu/download/download.html`
— `etherwake` (wake-on-lan) - budzenie przez sieć LAN
  Homepage: `http://www.scyld.com/expert/wake-on-lan.html`
  Debian: etherwake[605]

Narzędzia shell'owe

Podstawowym elementem każdego systemu operacyjnego (oczywiście poza samym jądrem i elementami mu towarzyszącymi …) jest interpretator poleceń (na ogół ten główny działa w trybie tekstowym, ale nawet jeżeli nie to tekstowy jest zawsze do naszej dyspozycji …) - shell. Najpopularniejszym tekstowym shell'em jest bash (podobnie jak pdksh oraz BSD'owski ash wywodzi się od sh, w odróżnieniu od shell'i wywodzących się z csh). Shell interpretuje komendy wprowadzane przez użytkownika, wykonuje programy a nawet jest skryptowym językiem programowania umożliwiającym automatyzację wielu czynności.

Wiele z podstawowych funkcji realizowanych jest przez niezależne polecenia (ich postać na ogół nie odróżnia się od komend wbudowanych w shell'a). Do takich należą zbiory podstawowych użytków tekstowych (grep, sed, less, czy textutils w skład którego wchodzą między innymi: wc, uniq, sort) i plikowych (fileutils w skład którego wchodzą między innymi: ls, cp, ln, mkdir, rm).

— `bash` - chyba najbardziej rozbudowana powłoka tekstowa …
  Homepage: `http://www.gnu.org/software/bash/`
— `zsh` - shell o składni typu sh przeznaczony głównie do pracy interaktywnej, posiada bogate i wygodne opcje konfiguracyjne
  Homepage: `http://www.zsh.org/`
— `fish` - shell przeznaczony głównie do pracy interaktywnej o nietypowej składni
  Homepage: `http://www.zsh.org/`
— `ipython` - interaktywny shell pythona przeznaczony głównie do debugowania ale dobrze sprawdzający się jako shell systemowy, zobacz ipython @ dug[606]
  Homepage: `http://ipython.scipy.org/`

— `pdksh` - kolejny shell wywodzący się z sh
  Homepage: http://web.cs.mun.ca/~michael/pdksh/
— `tcsh` - znany shell "typu C"
  Homepage: http://www.tcsh.org/
— `mc` - Midnight Commander - pełnoekranowy manager plików
  Homepage: http://www.ibiblio.org/mc/
— `screen` - menadżer ekranu - umożliwia utrzymywanie i zarzadzanie wieloma sesjami terminalowymi (w tym odłączanie i ponowne podłączanie odpalonych programów do terminala)
— `tmux` - menadżer ekranu - umożliwia utrzymywanie i zarzadzanie wieloma sesjami terminalowymi (w tym odłączanie i ponowne podłączanie odpalonych programów do terminala)
— `watch` - okresowe powtarzanie danej komendy z wyjściem pełnoekranowym

Przydatne skróty

Na koniec napiszę jeszcze o kilku przydatnych skrótach klawiszowych:

Ctrl +[r] - przeszukiwanie historii linii poleceń w bashu (i innych shellach opartych na readline), to samo do ponownego wyszukania,
lewy Alt +[Ctrl]+[numeryczny +] - kolejna skonfigurowana rozdzielczość w X'ach,
lewy Alt +[Ctrl]+[numeryczny -] - poprzednia skonfigurowana rozdzielczość w X'ach,
lewy Alt +[Ctrl]+[Esc] - zabójczy kursor w X'ach,
lewy Alt +[Ctrl]+[Backspace] - zabicie X'ów (w nowszych wersjach xorg działanie tej kombinacji może wymagać dodania `Option "DontZap" "false"` w sekcji "ServerFlags" xorg.conf),
lewy Alt +[lewy przycisk myszy] - przesuwanie okna w X'ach,
lewy Alt +[prawy przycisk myszy] - zmiana rozmiaru okna w X'ach,
lewy Alt +[środkowy przycisk myszy] - okno na wierzch/pod spód w X'ach,
lewy Alt +[F2] - okienko uruchom w KDE (pozwala także na "uruchomienie" URL czy też działań matematycznych),
Ctrl +[F*] - przełączanie pulpitów w KDE,
lewy Alt +[Ctrl]+[F*] - przełączanie terminali wirtualnych,
Scroll Lock - zatrzymanie/wznowienie przewijania konsoli (podobnie [Ctrl]+[s]/[Ctrl]+[q])

Z innych wygodnych "skrótów" warto wspomnie o środkowym i prawym przycisku myszy na ikonie maksymalizacji okna w KDE - powodują maksymalizację odpowiednio w pionie i poziomie

SSH, scp, sftp

Składnia poleceń SSH:
`ssh użytkownik@serwer` (następnie akceptujemy klucz kodowania (przy pierwszym połączeniu) i podajemy hasło) lub `ssh serwer` (następnie podajemy nazwę użytkownika, akceptujemy klucz kodowania (przy pierwszym połączeniu) i podajemy hasło)
`scp ścieżka-źródła ścieżka-celu` lub `scp -r ścieżka-źródła ścieżka-celu` (pozwala na rekurencyjne kopiowanie katalogów), gdzie ścieżka oznacza ścieżkę lokalną lub użytkownik@serwer: ścieżka-na-serwerze (następnie również akceptujemy klucz kodowania (przy pierwszym połączeniu) i podajemy hasło)
`sftp`

Przy pomocy ssh (bez scp) tez da się przesyłać pliki (w przypadku dużej ilości plików jest to metoda wydajniejsza, którą możemy także łatwo wzbogacić o kompresję):

— na host zdalny: `tar cf - pliki do przeslania | ssh user@host 'cd katalog/docelowy; tar xf -'`
— z hosta zdalnego: `ssh user@host 'cd katalog/docelowy; tar cf - pliki do przeslania' | tar xf -`

W przypadku gdy chcemy bezpośrednio poprzez ssh wywołać jakąś komendę interaktywną po zdalnej stronie (bez odpalania shell'a przydatna może być opcja `-t` wymuszająca utworzenie wirtualnego terminala - dzięki niej zadziała poprawnie `ssh 'sudo bash'` itp komendy.

W `~/.ssh/config` możemy definiować skrócone nazwy dla hostów wraz z określeniem opcji używanych przy logowaniu (np. port, użytkownik). Ponadto nazwy hostów podane w tym pliku będą rozwijane w ramach pa-

kietu "bash_completion" poprzez tab. Dodanie do tego pliku `HashKnownHosts no` powoduje że nazwy hostów w `~/.ssh/known_hosts` będą zapisywane w sposób jawny i również będą autouzupełniane przez "bash_completion".

W przypadku systemów Windows konieczna jest instalacja programów "Putty" (do połączeń SSH, tekstowe programy scp, sftp dostarczane z Putty noszą nazwy pscp i psftp) i/lub WinSCP (graficzny interfejs do SCP) i/lub "ssh secure shell client" (terminal SSH i graficzny interfejs do SFTP).

Generowanie kluczy SSH

SSH umożliwia także opcję nawiązywania połączeń bez potrzeby podawania hasła - wykorzystuje się do tego klucze asymetryczne. Aby je wygenerować korzystamy z komendy `ssh-keygen -t rsa` (możemy także wybrać inny typ klucza - dsa), jeżeli zależy nam na uniknięciu podawania hasła generujemy klucze niezabezpieczone hasłem (bezpieczniej jest używać kluczy z hasłem i `ssh-agent` aby podawać je tylko raz ...). Wygenerowane klucze znajdą się w `~/.ssh/id_rsa` - prywatny, `~/.ssh/id_rsa.pub` - publiczny. Klucz publiczny należy przekopiować na komputer z którym chcemy się łączyć (np[. przy pomocy scp) i dopisać do `~/.ssh/authorized_keys` - `cat id_rsa.pub >> ~/.ssh/authorized_keys`.

Możliwa jest także konwersja pomiędzy formatem kluczy openssh a ssh2: `ssh-keygen -e -f id_dsa.pub > id_dsa_ssh2.pub` i w drugą stronę: `ssh-keygen -i -f id_dsa_ssh2.pub > id_dsa.pub`

Montowanie SFTP jako systemu plików

Możliwe jest montowanie zdalnych zasobów SFTP jako lokalnego systemu plików. Wymaga to zainstalowania pakietu z sshfs oraz załadowania modułu fuse (mechanizm fs opartego na fuse zastąpił opisywany tu niegdyś shfs stanowiący moduł jądra). Montowanie wykonujemy przy pomocy polecenia `sshfs -o workaround=rename serwer:katalogpunkt_montowania` (dość istotna jest opcja "workaround=rename" umożliwiająca przenoszenie plików na takim systemie). Oczywiście użytkownik wykonujący to polecenie musi mieć prawa do `/dev/fuse` (należeć do grupy fuse). Warto na koniec też zwrócić uwagę iż możliwe jest korzystanie z standardowych kluczy generowanych przez `ssh-keygen` (i umieszczanych w $HOME/.ssh/ jako: authorized_keys , id_rsa , id_rsa.pub).

W przypadku KDE możliwe jest też korzystanie z zasobów SFTP bez ich montowania - wystarczy w Konqueror'ze wpisać sftp://uzytkownik@nazwa.serwera .

Tunelowanie SSH

Możliwości SSH nie kończą się na udostępnieniu zdalnego shella i możliwości bezpiecznego przesyłu plików, umożliwia on także zdalny dostęp do programów działających w trybie graficznym. Wymaga to jednak włączenia opcji przekazywania protokołu X'ów (w przypadku ssh z linii poleceń opcja -X) oraz działającego na maszynie z której się łączymy serwera X'ów (na maszynie docelowej takiego serwera może nie być w ogóle - musza być tylko stosowne biblioteki oraz program który chcemy odpalić). Po uzyskaniu połączenia shellowego z włączoną opcją przekazywania X'ów programy graficzne odpala się tak jak każdy inny program.

SSH umożliwia także tworzenie tuneli przekierowujących połączenia TCP na zadany port komputera z którego wykonujemy połączenie ssh do określonego adresu IP (wraz z portem) dostępnego z maszyny (serwera ssh) do której się łączymy (może być to np. prywatny adres IP dostępny tylko wewnątrz LANu): `ssh -L lokalny_port:mapowany_adres_ip:mapowany_port uzytkownik@serwer_ssh`. Możliwe jest też tunelowanie w drugą stronę (`ssh -R mapowany_port_serwera:adres_ip:port uzytkownik@serwer_ssh`) co umożliwia utworzenie (na serwerze ssh) portu nasłuchujacego który umożliwi połączenie do komputera (z którego zestawiony został tunel) do którego może nie być bezpośredniego dostępu (np. za firewallem); należy tutaj zaznaczyć iż domyślnie nasłuchiwanie na zadanym porcie odbywa się tylko lokalnie aby to zmienić należy przed tym portem określić adres nasłuchiwania (lub * dla wszystkich), aby było to skuteczne musi być włączona w konfiguracji serwera opcja "GatewayPorts". Umożliwia to także proste tworzenie swego rodzaju sieci VPN o bardzo dobrym zabezpieczeniu.

Inną intersującą opcją jest wykozystanie ssh w roli proxy którego jeden koniec mamy lokalnie, a drugi gdzieś daleko. Funkconalność taką zapewnia opcja -D w ssh i wykorzystanie lokalnego końca tunelu jako proxy typu SOCKS. Kolejną z ciekawych opcji jest `-w` pozwalająca na tworzenie urządzeń tunelowych, które mogą posłużyć do realizacji prawdziwego VPN.

Restricted SFTP/SSH

Niekiedy zachodzi potrzeba nałożenia pewnych ograniczeń na to co mogą robić użytkownicy którym dajemy dostęp poprzez ssh i pochodne.

Jednym z takich ograniczeń może być dostęp tylko przez klucze do jakiejś wskazanej komendy - wymaga to utworzenia użytkownika baz hasła (w /etc/shadow dajemy coś co na pewno nie jest hashem hasła) oraz utworzeniu wpisu w `~/.ssh/authorized_keys` postaci: `command="svnserve -t",no-port-forwarding,no-agent-forwarding,no-X11-forwarding,no-pty ssh-rsa __KLUCZ__ITD` (podana w przykładzie `svnserve -t` pozwala na dostęp do svn poprzez ssh - `svn ls svn+ssh://serwer/repo/path`).

Innym ograniczeniem jest dostęp tylko do SFTP bez możliwości zalogowania się do shella, uzyskać to można poprzez wpis w `/etc/passwd` postaci: `sftponly:x:UID:GID:SFTP uploader:HOME/PATH:/usr/lib/openssh/sftp-server`. Jeżeli chemy udostępnic także inne protokoły kopiowania plików - takie jak np. rsync możemy skorzystać także z rssh[607].

Aby użytkownik nie tylko nie miał dostępu do shella ale także nie mógł wyjść poza swój katalog trzeba zastosować chroot, współczesne wersje OpenSSH bardzo ułatwiają budowę takiego środowiska należy w konfiguracji SSH umieścić następujące wpisy (polecam zwrócić uwagę na informacje w komentarzach):

```
Subsystem sftp internal-sftp
# zamiast Subsystem sftp /usr/lib/openssh/sftp-server
Match User ftplim1
        ForceCommand internal-sftp
        # należy też ustawić "Subsystem sftp" na internal-sftp
        X11Forwarding no
        AllowTcpForwarding no
        ChrootDirectory /var/chroot-sftp/%u
        # %u podmieniane jest na nazwę użytkownika, dzięki czemu można zamiast Match User
↪    użyć Match Group
        # w passwd $HOME musi wskazywać na coś nadrzędnego $SHELL może być /dev/null
        # jeżeli nie chcemy robić go global-writable to wewnątrz niego powinien być
↪    katalog który należy do ftplim1
```

Wpisanie `session optional pam_lastlog.so` w `/etc/pam.d/sshd` spowoduje logowaniw wszystkich połączeń ssh (np. połączeń sftp) w pliku wtmp (podobnie można uczynić z /etc/pam.d/su). Natomiast jeżeli występuje problem z starszymi klientami ssh (nieprzyjmowanie poprawnego hasła) to warto w konfiguracji tego serwera ustawić opcję ”PasswordAuthentication yes”.

Podstawy ”życia” w Debianie

Debian[608] jest to wolny system operacyjny, zawierający bardzo dużo oprogramowania, od innych dystrybucji odróżnia go położenie bardzo dużego nacisku na kwestie wolności poszczególnych składników oraz posiadanie wersji opartych na różnych jądrach (Linux, Hurd, BSD). W standardowej dystrybucji znajdziemy większość opisywanych wyżej programów. Należy jednak zwrócić uwagę iż w pojedynczej paczce .deb może być kilka programów (np. procmail zawiera formail; w takich wypadkach przydatna jest wyszukiwarka wewnątrz pakietowa - `http://packages.debian.org/`, gdyż komendy włączające program są na ogół zgodne z jego nazwą …), jak również jeden pakiet może być podzielony na bardzo wiele paczek .deb na ogół (ale nie zawsze) powiązanych zależnościami (np. KDE). Jednak niektórych wymienionych wyżej programów (cinelerra) nie ma (jeszcze …) w oficjalnej dystrybucji, stosownych paczek warto dodać do `/etc/apt/sources.list`. Warto także zwrócić uwagę na snapshot.debian.net[609] przechowujący starsze wersje pakietów (może być bardzo przydatny gdy używając Sid'a dorobimy się zabugowanej wersji kluczowego dla nas pakietu).

Instalacja oprogramowania

Polecam zakończenie działania programu dokonującego podstawowej konfiguracji systemu (automatycznie uruchamianego przy pierwszym starcie po ewentualnym wybraniu dodatkowych źródeł apta - w przypadku dobrego łącza do Internetu warto wybrać źródła HTTP lub FTP). System ten posiada na tyle wygodny system zarządzania

pakietami (chyba najlepszy z jakim się spotkałem), że późniejsze dodanie potrzebnego oprogramowania nie będzie problemem. Poniżej omówię kilka poleceń i programów związanych z zarządzaniem pakietami w Debianie.

aptitude - najbardziej wysokopoziomowy i "user-friendly" (z standardowych) program do zarządzania pakietami, mogący pracującować w tzw. "trybie pełnoekranowym". Do najważniejszych jego poleceń zaliczyć należy {podane w postaci: "znak wywołujący akcje w trybie pełnoekranowym" - "opis" ("komenda trybu linii poleceń")}:

+ - instaluje wybrany pakiet (install)

- - odinstalowywuje wybrany pakiet (remove)

_ - wykasowywuje (w raz z plikami konfiguracyjnymi) wybrany pakiet (purge)

L - reinstaluje wybrany pakiet

R - rekonfiguruje wybrany pakiet {odpowiada wywołaniu: dpkg-reconfigure nazwa_pakietu}

= - wstrzymuje aktualizację wybranego pakietu (hold)

I - instaluje "pojedyńczo" (bez spełniania zależności) wybrany pakiet

M - dodaje znacznik automatycznej instalacji (markauto)

m - usuwa znacznik automatycznej instalacji (unmarkauto)

/ - wyszukiwanie pakietu (standardowo według nazwy, aby szukać w opisie `~d` szukany_tekst, aby znaleźć niedokasowane `~c` lub nieoficjalne `!~Odebian`, aby szukać tylko w zainstalowanych `~i`)

\ - wyszukiwanie wstecz

n - następny wyszukany pakiet

N - poprzedni wyszukany pakiet

l - filtr pakietów (np. `?any-version(~i !~Astable !~Atesting)` zawęzi listę wyświetlanych pakietów tylko do tych których zainstalowana wersja jest różna od wersji z stable i testing)

d - wyświetla zależności danego pakietu

r - wyświetla zależności od pakietu

i - wyświetla informacje o pakiecie

g - wykonuje zaplanowane akcje

q - wychodzi z aktualnego okna bądź z programu

? - pomoc

jednym z mniej znanych, a bardzo przydatnych wywołań jest `aptitude -F '%c%a%M %p %V' search '~Rdepends:nazwa_pakietu'` wypisujące pakietów wymaganych itp dla danego

więcej: menu Pomoc w programie i `man 8 aptitude`

Warto także zwrócić uwage na plik konfiguracyjny aptitude (i apt'a) - `/etc/apt/apt.conf` (lub jego odpowiedniki w katalogu domowym) - umożliwia on m.in. wyłączenie instalowania pakietów polecanych:

```
# nie trzyma polecanych i sugerowanych jak zaleznoŚci
APT::AutoRemove::RecommendsImportant "false";
APT::AutoRemove::SuggestsImportant "false";
# nie instaluje automatycznie olecanych i sugerowanych
APT::Install-Recommends "false";
```

```
APT::Install-Suggests "false";
# wyłączenie wyszukiwania przyrostowego w aptitude
aptitude::UI::Incremental-Search "false";
# wyłączenie pobierania aktualizacji listy pakietów jako pdiff
# (polecam jeżeli stosunkowo rzadko aktualizujemy listy pakietów np. na testingu)
Acquire::PDiffs "false";
```

apt-get - program do pobierania i instalowania pakietów ...

apt-file - program umożliwiający wyszukiwanie pakietu zawierającego plik o podanej nazwie lub ścieżce (w odróżnieniu od `dpkg -S` nie ogranicza się tylko do zainstalowanych pakietów) - `apt-file update` - aktualizacja informacji o pakietach, `apt-file search test` - wyszukanie pakietów zawierających plik test

apt-build - program ułatwiający kompilowanie Debiana ze źródeł, celem optymalizacji

dpkg - potężne niskopoziomowe narządzie do zarządzania pakietami. Do najważniejszych opcji zaliczyć należy:

-i nazwa_pakietu [...] - instaluje pakiet(y)

-l wzorzec_nazwy_pakietu [...] - wypisuje informacje o stanie pakietów (w szczególności czy jest zainstalowany)

-L nazwa_pakietu [...] - lista plików zawartych w pakietach

-S sciezka_do_pliku - lista pakietów zawierających plik

–get-selections - lista zainstalowanych pakietów wraz z statusem - aktualnie zainstalowany (install) lub usunięty (deinstall)

warto zwrócić uwagę na możliwość zapisania listy zainstalowanych pakietów do pliku - `dpkg --get-selections > pakiety.txt` oraz zaznaczenia do instalacji pakietów zapisanych w tak wygenerowanej liście - `dpkg --set-selections < pakiety.txt`

więcej - tradycyjnie: `man 8 dpkg`, `dpkg --help | less`

Zachęcam do zapoznania się także z pozostałymi programami z rodziny apt-* (niektóre z nich to osobne paczki, wymagające doinstalowania) - umożliwiają one m.in. budowę systemu Debian ze źródeł (jak Gentoo ...) oraz programem aptsh (shell do zarządzania pakietami, podobnie jak jest w PLD).

Niekiedy przydatna może być sztuczka z fałszywym zainstalowaniem paczki. Operacja taka sprowadza się do:

— dodaniu do `/var/lib/dpkg/status`:
  `Package: NAZWA_PACZKI`
  `Status: install ok installed`
  `Version: 999.999.999`
— utworzeniu pliku do `/lib/dpkg/info/NAZWA_PACZKI.list`

Struktura katalogów i pliki konfiguracyjne

Na wstępie warto napisać jeszcze parę słów o strukturze katalogowej w Debianie (a poniekąd też i innych ”Unixach”) oraz kilka ogólnych uwag o plikach konfiguracyjnych i ich edycji. Katalog główny (korzeń, root) oznaczany jest przez / w nim znajdują się m.in. podkatalogi: `/bin` (pliki wykonywalne podstawowych programów), `/sbin` (pliki wykonywalne podstawowych programów administracyjnych), `/lib` (podstawowe biblioteki), `/etc` (pliki konfiguracyjne), `/var` (dane trwałe stacji - np. kolejka wydruków czy też pocztowa, logi systemowe - podkatalog `/var/log`), `/tmp` (pliki tymczasowe, usuwane przy starcie), `/usr/bin` (pliki wykonywalne pozostałych programów), `/usr/sbin` (pliki wykonywalne pozostałych programów administracyjnych), `/usr/lib` (pozostałe biblioteki), `/usr/include` (pliki nagłówkowe C/C++), `/usr/share` (pliki dodatkowe - danych ... zainstalowanych programów), `/usr/src` (źródła), `/usr/local` (podobnie jak `/usr` ale wykorzystywane głównie dla programów dokompilowywanych). Zobacz w sieci: Filesystem Hierarchy Standard[610].

Warto tutaj także wspomnieć iż właśnie w (pozornie może niezbyt interesującym) katalogu /var przechowywana jest konfiguracja dotycząca zainstalowanych paczek *.deb. Warto od czasu do czasu zadbać o jej zarchiwizowanie np. przy pomocy `tar -czf /etc/pakiety-stan.tgz /var/lib/dpkg/status`

/var/lib/dpkg/statoverride /var/lib/dpkg/diversions /var/lib/dpkg/alternatives /var/lib/dpkg/info /var/lib/apt/extended_states /var/lib/aptitude/pkgstates.

Jak już wspomniałem pliki konfiguracyjne systemu znajdują się w katalogu /etc i jego podkatalogach (pliki konfiguracyjne użytkowników to ukryte (zaczynające się od kropki) pliki i katalogi w jego katalogu domowym). Zasadniczo każda aplikacja ma własny format plików konfiguracyjnych, dlatego też trudno podać jakieś ogólne informacje o nich i ich edycji … informacji tych należy szukać w dokumentacji danego programu, także stronach mauala opisujących dane pliki (rozdział 5), bardzo przydatne gdy chcemy cos zrobić, a za bardzo nie wiemy jak jest Google[611] … . Można jednak powiedzieć że generalnie są to pliki tekstowe, niekiedy z podziałem na sekcje, najczęściej oparte o zasadzie że w kolejnych liniach są podawane ciągi: `klucz wartosc`, komentarzami na ogół są całe linie zaczynające się od #.

GRUB

Po zakończeniu procesu inicjalizacji sprzętu i testów rozruchowych BIOS ładuje do pamięci kod znajdujący się w pierwszym sektorze dysku twardego (sektorze rozruchowym[612] rozpoczynającym się od adresu zerowego) i uruchamia go (przekazuje do niego kontrolę). W przypadku sektorów rozruchowych typu MBR[613] (i kompatybilnych z nim), kod ten może liczyć maksymalnie 446 bajtów (gdyż na kolejnych pozycjach znajduje się tablica partycji) i jego zadaniem jest załadowanie i uruchomienie pozostałej częsci programu rozruchowego[614] (np. może użyć programu w partycji oznaczonej jako bootowalna).

GRUB[615] jest jednym z najpopularniejszych w środowisku linuxowym boot menagerów / boot loaderów. Jego wykonywalna przy starcie komputera wersja składa się z 3 części:

- stage1 (boot.img) - ma za zadanie uruchomić stage1.5 (zawiera jego położenie na dysku)
- stage1.5 (core.img) - zawiera moduły pozwalające na dostęp do systemu plików zawierającego stage2 oraz informację o jego położeniu (dysk, partycja, ścieżka)
- stage2 (/boot/grub) - zawiera wszystkie dodatkowe moduły oraz konfigurację

Może on być zainstalowany zarówno w MBR, jak i w bootowalnej partycji. W tym pierwszym przypadku w MBR wgrywany jest stage1, natomiast stage1.5 w zależności od używanego typu tablicy partycji znajduje się albo tuż za MBR - w przerwie pomiędzy MBR a pierwszą partycją (dla partycji MBR/msdos), albo w dedykowanej partycji BIOS boot partition[616] (w przypadku GPT).

W przypadku komputerów opartych na UEFI[617] firmware odpowiedzialny jest za zinterpretowanie tablicy partycji (GPT[618]) i załadowanie progragramu rozruchowego z pliku znajdującego się na specjalnej partycji EFI (EFI System partition[619]) z systemem plików FAT32[620]. GRUB może pełnic funkcję tak uruchamianego boot loadera (w tym przypadku całość potrzebnego kodu stage1 i stage1.5 znajduje się w pliku na tej partycji).

Zadaniem GRUBa jest umożliwienie wyboru, załadowanie i uruchomienie jądra systemu operacyjnego z zadanymi opcjami. Jego działanie kończy się w momencie gdy jądro przejmuje sterowanie. Zobacz także w Sieci: GRUB[621], MBR[622], GPT[623], UEFI[624].

Konfiguracja

Współcześnie konfiguracja GRUB'a jest generowana automatycznie w oparciu o /etc/grub.d/ i /etc/default/grub przez grub-mkconfig w oparciu o wykrywanie typów dysków na których jest zainstalowany system (raid, lvm, typ systemu plików, ...) oraz unikalne identyfikatory dysków, partycji, itd. Mimo to warto znać przynajmniej podstawy działania grub'a, jego komend, aby być wstanie coś zrobić gdy zamiast menu startowego ujrzymy tylko konsolę awaryjną. Poniżej zamieszczam przykładowy plik konfiguracyjny /boot/grub/grub.cfg z komentarzami opisującymi zanczenie różnych poleceń i opcji.

```
# oficjalna dokumentacja: http://www.gnu.org/software/grub/manual/grub.html
# korzystanie z konsoli na porcie szeregowym
#serial --speed=115200 --unit=1 --word=8 --parity=no --stop=1
#terminal_input serial console
#terminal_output serial console
# za to na którym RS jest grub odpowiada --unit w serial
#
# korespondujące opcje uruchamiania dla Linuxa:
#   module    /boot/vmlinuz [...] console=tty0 console=ttyS1,115200n8
# korespondujące opcje uruchamiania dla XENa:
#   multiboot /boot/xen.gz  [...] com2=115200,8n1 vga=mode-0x0319 console=com2,vga
```

```
#   module    /boot/vmlinuz [...] console=tty0 console=hvc0
#
# w opcjach kernela szczególną rolę odgrywa ostatni podany terminal (ostatnia opcja
 ↪  console)
# jest on związany z /dev/console i na nim pokazywany jest init, ponadto należy
 ↪  uruchamiać getty
# na odpowiednim urządzeniu poprzez wpis w inittab np.:
#   T0:23:respawn:/sbin/getty -L /dev/ttyS1 115200 vt100
# lub (dla wariantu XENowego, UWAGA: w przykładzie tylko runlevel 4 i 5):
#   T1:45:respawn:/sbin/getty 38400 hvc0
# timeout w sekundach dla domyślnej pozycji menu
set timeout=60
set default="0"
# ładowanie modułów - typ tablicy partycji
insmod part_gpt
#insmod part_msdos
# ładowanie modułów - RAID
insmod mdraid1x # mdraid metadata 1.x
#insmod mdraid09 # mdraid metadata 0.9
#insmod raid5rec # faulty RAID4/5
#insmod raid6rec # faulty RAID6
# ładowanie modułów - LVM
insmod lvm
# ładowanie modułów - File System
insmod xfs
# ustawienie root'a
set root='lvm0-root'
menuentry 'Debian GNU/Linux' {
        # obraz jądra Linuxa z przekazywanymi do systemu opcjami
        linux   /boot/vmlinuz-3.12-trunk-amd64 root=/dev/mapper/lvm0-root ro rootdelay=3
 ↪  quiet
        # obraz initrd
        initrd  /boot/initrd.img-3.12-trunk-amd64
}
menuentry "SBM Boot Manager" {
        linux16  /boot/extra/memdisk floppy
        # memdisk z pakietu syslinux-common (/usr/lib/syslinux/memdisk)
        initrd16 /boot/extra/sbootmgr.dsk
        # SBM Boot Manager (http://btmgr.sourceforge.net/about.html)
        # z http://ftp.lanet.lv/ftp/mirror/Slackware/isolinux/sbootmgr/
}
```

Start systemu - initrd, init i pliki startowe

Start systemu rozpoczyna się od załadowania do pamięci obrazu jądra wraz z parametrami oraz (opcjonalnie) initrd i przekazania kontroli do jądra przez program rozruchowy (np. GRUB, LILO). W przypadku korzystania z initrd obraz ten przekształcany jest na RAM-dysk w trybie zapisu-odczytu i montowany jako rootfs z którego uruchamiany jest `/sbin/init`. Po zakończeniu jego działania (lub od razu gdy nie używamy initrd) uruchamiany jest program wskazany w opcji `init=` jądra (domyślnie typowo `/sbin/init`) z rootfs wskazanego w opcji `root=` jądra.

Zawartość obrazu rozruchowego można podejrzeć poprzez jego rozpakowanie komendą: `mkdir -p /tmp/initrd && cd /tmp/initrd && gzip -dc < /initrd.img | cpio -i` (jednak ponowne spakowanie go nie przynosi pożądanych rezultatów, ponadto współczesnych obrazów nie da się już montować przez mount -o loop). Celem dostosowywania initrd do własnych potrzeb polecam przyjrzenie się strukturze katalogów `/usr/share/initramfs-tools/` i `/etc/initramfs-tools/`. W szczególności wart przeanalizowania jest skrypt `/usr/share/initramfs-tools/init`, który jest uruchamiany zaraz po odpaleniu obrazu startowego (jest kopiowany przy tworzeniu initrd.img na /init). Po wykonaniu stosownych zmian w tych plikach obraz można zbudować poprzez wywołanie `mkinitramfs` lub wygodniejszego `update-initramfs -u`.

Część z opcji przekazywanych jako parametry jądra jest obsługiwana przez initrd - wybrane z tych opcji:

— `break=top|modules|premount|mount|mountroot|bottom|init` - przerywa initrd (i uruchamia powłokę busybox) w wskazanym momencie (w initramfs-tools 0.92o jest błąd powodujący iż break-point init nie działa - przed wywołaniem maybe_break init jest wstawione unset break), dalsze bootowanie można wznowić Ctrl+D
— `rootdelay=xxx` - wstawia opóźnienie xxx sekund przed próbą montowania głównego systemu plików (pomiędzy inicjalizacją urządzeń fizycznych a startem skryptów local - tworzących urządzenia md, lvm itp), jej uzycie może być konieczne w przypadku korzystania z rootfs na lvm, soft-raid itp
— `rootflags=` - podaje flagi montowania głównego systemu plików (przekazane do opcji -o komendy mount)
— `boot=` - określa jakie skrypty celem zamontowania root fs ma zainkludować initrd (domyślnie local), boot=xxx nakazuje wykonanie skryptu /scripts/xxx (np. /scripts/nfs) przed montowaniem głównego systemu plików
— `debug` - zwiększa głośność initrd
— `blacklist=` - rozdzielana przecinkami lista modułów których ładowania chcemy zabronić na etapie initrd
— `init=` - określa jaki program ma być wykonany zaraz po zakończeniu initrd

Więcej opcji linii poleceń jądra znaleźć można w `man bootparam` oraz `man init`. Zaznaczyć tu należy iż zgodnie z `man bootparam` parametry postaci nazwa=wartosc nie rozpoznane jako parametry jądra są przekazywane jako zmienne środowiskowe, obecnie trafiają one w tej postaci do skryptów wywoływanych przez init (jednak zdarzało się że gdzieś ginęły). Natomiast same procesy związane z logowaniem użytkownika nie przekazują tych zmiennych do środowiska obecnego po zalogowaniu, zatem aby z nich skorzystać konieczne jest przetwarzanie `/proc/cmdline` (ze względów na nie pewność co do ustawienia środowiska przez init warto tak postępować także w skryptach init). Można to realizować w następujący sposób ``ZMIENNA=`awk 'BEGIN {RS="[ \n]"; FS="=";} $1=="ZMIENNA" {print $2}' /proc/cmdline` `` lub:

```
for tmp in $(cat /proc/cmdline); do
	case $tmp in
		ZMIENNA=*)
			ZMIENNA=${tmp#ZMIENNA=}
			;;
	esac
done
```

Jak już wspomniano po zakończeniuinitrd typowo uruchamiany jest standardowy init (polecam zapoznanie się z plikiem konfiguracyjnym - /etc/initab), oferuje on 7 poziomów działania (0 - wyłączenie, 1 - pojedynczy użytkownik (single), 2...5 - standardowe (najczęściej używany 2), 6 - restart). Program ten podczas zmiany poziomu rozruchu przetwarza skrypty startowe z odpowiedniego katalogu /etc/rcX.d (X- poziom rozruchu na który przechodzi init), w katalogach tych znajdują się najczęściej dowiązania symboliczne do skryptów umieszczonych w /etc/init.d. Pliki w katalogach /etc/rcX.d mają zawsze nazwy postaci YZZxxxx, gdzie Y to S gdy dany skrypt wykonywany ma być z parametrem ”start” albo K gdy z parametrem ”stop”, ZZ to dwucyfrowy numer decydujący o kolejności wykonania, xxxx - nazwa skryptu. Przy starcie systemu najpierw wykonywane są skrypty z /etc/rcS.d a następnie zadanego poziomu (zazwyczaj /etc/rc2.d, rzadziej z /etc/rc1.d - tu na ogół i tak są same ”stop”, a jeszcze rzadziej z innych).

Standardowo skrypty (będące na ogół zwykłymi skryptami powłoki) mają dość złożoną strukturę - zależność działania od argumentu (start, stop, restart, ...), kontrola prawa wykonywalności uruchamianego programu; nic jednak nie stoi na przeszkodzie aby umieścić tam dowolne inne (dużo prostsze ;-)) skrypty. Jest też programik do zarządzania rozruchem systemu - `update-rc.d`.

SYSTEMD

Współcześnie funkcję klasycznego init'a przejmuje systemd. W odróżnieniu od klasycznego podejścia nie opiera się on na skryptach umieszczanych w `/etc/init.d` i linkowanych do odpowiedniego `/etc/rc*.d` a na opisach usług w postaci plików konfiguracyjnych typu `.desktop` / `.ini`, jednak obsługuje też te pierwsze.

Do zarządzania plikami usług (listowania dostępnych, włączania/wyłączania autostartu, blokowania, itd) służą m.in. następujące polecenia programu `systemctl`:

```
systemctl  list-unit-files
systemctl  enable|disable  NAZWA_USLUGI
        ls /etc/systemd/system/*.target.wants/
systemctl  mask|unmask  NAZWA_USLUGI
systemctl  show|list-dependencies  NAZWA_USLUGI
systemctl  cat|edit  NAZWA_USLUGI
```

Poziomy startu, czyli to jakie usługi kiedy mają byćn uruchomione (z wyłączeniem rzeczy wynikłych z zależności - są one uruychamiane automatycznie na podstawie zależności zdefiniowanych w pliku usługi) opisywane są linkami w `/etc/systemd/system/*.target.wants/`

Do zarządzania aktualnym stanem usługi (jej startowania, zatrzymywania itd) a także wyświetlania timerów służy również program `systemctl` z następującymi poleceniami:

```
systemctl  list-units|list-timers
systemctl  status|start|stop|restart|reload  NAZWA_USLUGI
```

Jak wspomniano systemd jest kompatybilny z klasycznymi sktyptami startowymi - automatycznie generuje dla nich pliki konfiguracyjne usług i umieszcza je w `/run/systemd/generator.late/` a linki do usług związanych z danym poziomem uruchomienia w `/run/systemd/generator.late/*.target.wants/`.

Zobacz w Sieci: Systemd[625], Systemd - FAQ[626], Systemd - Timers[627], Systemd - User[628], Getting Started With systemd on Debian Jessie[629], How To Use Systemctl to Manage Systemd Services and Units[630], Can systemd replace monit?[631].

Moduły - ładowanie, konfiguracja, …

Jądro linuxa od dłuższego już czasu ma zmodularyzowana budowę i wiele sterowników (urządzeń, protokołów, …) oraz innych funkcji zawartych jest w osobnych modułach. Moduły te mogą być ładowane podczas startu systemu (`/etc/init.d/module-init-tools`) bądź być ładowane dynamicznie lub ręcznie podczas działania systemu. Plikiem określającym jakie moduły i z jakimi parametrami chcemy załadować podczas startu jest `/etc/modules`, gdzie w każdej linii nie zaczętej od # pierwszy wyraz to nazwa modułu a wszystko następne to parametry.

Za konfigurację modułów (oprócz wspomnianego pliku z listą ładowanych modułów) odpowiadają również pliki w katalogu `/etc/modprobe.d` (wszystkie pliki w tym katalogu są równoważne). W skrypcie startowym modułów znajduje się polecenie budujące drzewo zależności (depmod), jeżeli zależy nam na przyspieszeniu startu komputera możemy je wyłączyć i uruchamiać ręcznie po zmianie, dodaniu, … jakiś modułów, bądź modyfikacji ich parametrów w wspomnianych wyżej ktalogach.

Trzeba także zaznaczyć że w nowszych jądrach gdzie wykorzystywany jest udev, większość modułów (domyślnie) ładowana będzie automatycznie - dlatego ważniejsza jest konfiguracja `/etc/modprobe*` niż `/etc/modules`. Trzeba także zwrócić uwagę na konfigurację `/etc/udev/*`.

Moduły są ładowane na etapie initrd (wtedy aby jakiekolwiek zmiany konfiguracyjne mogły odnieść skutek należy przebudować initrd) lub na etapie normalnego init'a - poprzez udev (`/etc/rcS.d/S03udev`) lub na podstawie wspomnianego `/etc/modules` (`/etc/rcS.d/S20module-init-tools`). Oczywiście inne skrypty startowe mogą ładować swoje moduły, jeżeli robią to np. przez insmod (a nie modprobe) to wszelkie opcje z konfiguracji udev i modprobe są ignorowane.

UDEV

udev jest mechcanizmem służącym do dynamicznego zarządzania zawartością katalogu `/dev`, jego konfiguracja znajduje się w katalogu `/etc/udev` (osobiście polecam swoje wpisy umieszczać na początku pliku udev.rules, kolejność wykonywania reguł określana jest przez porządek linków w `/etc/udev/rules.d` i jest ona istotna) i umożliwia m.in. (oczywiście to tylko parę przykładów zastosowań konfiguracji udev):

- Określenie numeracji kart sieciowych na podstawie adresów sprzętowych: `KERNEL=="eth*", SYSFS{address}=="mac:adres:wybranej:karty:sieciowej:", NAME="eth0"` (jeżeli komputer posiada dwie karty sieciowe a w konfiguracji udev nie określimy ich kolejności numeracja ich może się zmienić po restarcie - nawet jeżeli używają różnych modułów jako sterowników).
- Określanie własnych nazw urządzeń: np. `BUS=="scsi", SYSFS{model}=="USB Storage-CFC ", NAME{all_partitions}="flash/cf"` określa że wszystkie partycje (tak na zapas) karty CF która może być umieszczona w czytniku USB Mass Device Storage (widzianym jako urządzenie magistrali scsi) o nazwie modelu "USB Storage-CFC " mają mieć nazwy `/dev/flash/cfX`, gdzie X jest numerem partycji.
- Konfigurację dowiązań symbolicznych do urządzeń znajdujących się w katalogu `/dev`: np. `KERNEL=="radio0", SYMLINK+="radio"` nakazuje utworzenie dowiązania symbolicznego `/dev/radio` do `/dev/radio0`.
- Oba powyższe w jednym - np. `BUS=="serio", KERNEL=="mouse*", DRIVER=="psmouse", SYMLINK="input/mouse0", NAME="input/psmouse"` nakazuje myszkę PS/2 podłączyć do `/dev/input/psmouse` oraz utworzyć do niej dowiązanie `/dev/input/mouse0`
- Konfigurować uprawnienia dostępu do urządzeń - poprzez dodanie do wpisu tworzącego urządzenie `OWNER="wlasciciel", GROUP="grupa", MODE="prawa_numerycznie"` lub interesującego nas fragmentu tego wyrażenia.

Przy tworzeniu i testowaniu reguł udev'a pomocna może być komenda udevadm, która umożliwia testowanie reguł dla danego urządzenia (określonego ścieżką względem `/sys`) - np. `udevadm test /devices/platform/ipmi_si.0/ipmi/ipmi0`, uzyskanie informacji o istniejącym urzadzeniu w dev - np. `udevadm info --query=all --name=ipmi0`, czy też przeładowanie i wykonanie reguł na działającym systemie (w celu np. zmiany nazw interfejsów sieciowych):

```
udevadm control --reload-rules
udevadm trigger --action=add --verbose --subsystem-match=net
```

Udev do identyfikacji urządzeń oprócz czytania informacji z `/sys/` korzysta z różnych pomocniczych programów są to m.in.:

- `/sbin/blkid /dev/sda` - informacje dotyczące filesystemu, (tablicy partycji) - UUID, LABEL, etc
- `/lib/udev/ata_id --export  /dev/sda1` - informacje dotyczące urządzenia ATA - MODEL, SERIAL, WWN
- `/lib/udev/scsi_id --export --whitelisted /dev/sda1` - informacje dotyczące urządzenia SCSI - MODEL, SERIAL, WWN
- `/lib/udev/v4l_id /dev/video0` - informacje dotyczące urządzenia V4L

Zazwyczj całkiem dobrym miejscem na umieszczanie swoich wpisów jest `/etc/udev/udev.rules`. Na koniec wspomnę że udev'owi nie zawsze udaje się ładować wszystko autmagicznie i np. aby mieć `/dev/lp0` należy załadować (np. przez `/etc/modules`) moduł `lp`. Podobnie warto tam dopisać `ipv6`. Podobną rolę jak wpisy udev mogą pełnić w niektórych przypadkach parametry modułów określające z jakim numerem ma występować urządzenie (tak jest np. w modułach v4l) - możemy je podać w konfiguracji modprobe. Z kolei nieskuteczną drogą jest określanie numerów urządzeń poprzez wpisy alias w konfiguracji modprobe.

Zobacz też: Automontowanie systemów plików (automount, autofs)[632].

Tablica partycji GPT

GUID Partition Table[633] (niekiedy określany także jako tablica partycji EFI) jest jednym z kilku stosowanych typów tablic partycji (należy tutaj wspomnieć o tablicach typu disklabel[634] charakterystycznych dla rodziny BSD, partycjach typu x86 (msdos, fdisk, mbr[635]) charakterystycznych dla platformy PC, typu APM[636] charakterystycznych dla Apple). W porównaniu do wymienionych konkurentów jest rozwiązaniem dość nowym i posiadającym zalety wobec nich (obsługa dużych partycji, bezpośrednia - bez zagnieżdżania bsd disklabel - obsługa systemów BSD, obsługa do 128 partycji, itd).

Do tworzenia tego typu partycji pod linuxem możemy posłużyć się GNU parted (niestety fdisk ich nie wspiera):

```
# tworzymy tablicę partycji typu gpt
parted $DEV "mklabel gpt"
# tworzy partycję która posłuży do wgrania gruba
# należy zaznaczyć iż nie jest to partycja /boot
# jest to surowe (bez filesystemu) miejsce na dysku gdzie
# zostanie wgrany fragment gruba normalnie wgrywany zaraz za MBR
parted $DEV "mkpart grub 0 2MB";
# ustawiamy dla tej partycji flagę "GRUB BIOS partition"
parted $DEV "set 1 bios_grub on"
# resztę dysku możemy podzielić wg uznania
# w tym przykładzie robimy jedną dużą partycję o nazwie raid
parted $DEV "mkpart raid 2MB 100%";
```

Tak zainstalowany GRUB2 radzi sobie bez problemu z partycjami GPT, macierzami mdadm, wolumenami LVM itd.

RAID i LVM - bezpieczniejsze, większe i bardziej elastyczne partycje dyskowe

Linux oferuje dwie przydatne technologie dotyczące zarządzania pamięciami masowymi - jest to programowy RAID[637] oraz woluminy logiczne LVM[638]. RAID umożliwia realizację różnych form mirroringu mających na celu zabezpieczenie przed utratą danych, a także uzyskiwanie większych przestrzeni złożonych z kilku dysków. LVM służy umożliwieniu bardziej elastycznego podziału dysku oraz uzyskania logicznych partycji złożonych z wielu różnych

fragmentów dysków fizycznych. Również główny system plików (/) może korzystać z tych dobrodziejstw, należy tylko pamiętać aby gdy korzystamy z LVM bądź RAID'a innego niż RAID1 zostawić zwykłą partycję na /boot (pliki bootloadera i obrazy jądra).

```
    # tworzymy RAID1 dla partycji /
    mdadm -C -v /dev/md0 --level=1 -n 2 /dev/sda1 /dev/sdb1
    # tworzymy zdegradowany RAID1 dla dwóch partycji na których będzie /home
    mdadm -C -v /dev/md1 --level=1 -n 2 /dev/sda3 missing
    mdadm -C -v /dev/md2 --level=1 -n 2 /dev/sdb3 missing
    # tworzymy volumeny fizyczne na urządzeniach RAID dla potrzeb LVM
    pvcreate /dev/md1
    pvcreate /dev/md2
    # tworzymy grupę voluminów dla LVM
    vgcreate lvm0 /dev/md1
    # dodajemy volumen fizyczny do grupy
    vgextend lvm0 /dev/md2
    # można też usunąć przy pomocy:
    # vgreduce lvm0 /dev/md2
    # tworzenie volumenu logicznego o zadanej wielkości i nazwie w ramch podanej
↪  grupy
    # będzie z nim związane urządzenie /dev/lvm0/home
    lvcreate -L 25G -n home lvm0
    # ogladamy to co żesmy stworzyli
    pvdisplay
    vgdisplay
    lvdisplay
    # powiększamy volumen logiczny
    lvextend  -L +1GB  /dev/sys/homes
    # powiększamy system plików, np.
    #  xfs_growfs /home
    #  resize2fs /dev/lvm0/home
    #  btrfs filesystem resize max /home
    #  # więcej poleceń btrfs-owych:
↪  https://btrfs.wiki.kernel.org/index.php/Btrfs%28command%29
    # uzupełniamy nasz zdegradowany raid
    /sbin/mdadm -a /dev/md1 /dev/sdc1
    /sbin/mdadm -a /dev/md2 /dev/sdc2
```

Do kasowania macierzy możemy posłużyc się komendą: `mdadm -S --zero-superblock /dev/md14; mdadm -S /dev/md11`

Warto zachęcać LVM do używania identyfikatorów dysków w tym celu: w `/etc/mdadm/mdadm.conf` umieszczamy wpis `DEVICE /dev/disk/by-id/*`, macierz budujemy (w tym przykładzie raid 6 na 8 dyskach) np. `mdadm -C -v --auto=mdp --level=raid6 --raid-devices=8 /dev/disk/by-id/scsi-*part2` i w oparciu o wynik `mdadm --detail --scan` edytujemy `/etc/mdadm/mdadm.conf`

W zasadzie LVM sam się konfiguruje przy uruchamianiu jądra, ale niekiedy może zajść potrzeba ręcznego uruchamiania LVM w trakcie startu systemu. Procedura takiej operacji wygląda następująco:

1. załadowanie modułów od devicemaper `dm-mod` i od lvm `lvm*` (jeżeli jest jako moduły)
2. jeżeli udev nie potworzył urządzeń to tworzymy ręcznie:
   ```
   mknod --mode=600 /dev/lvm c 109 0
   mknod --mode=600 /dev/mapper/control c 10 62
   # numery możemy odczytać z /proc/device:
   #  major = misc
   #  minor = device-mapper
   ```
3. wykonanie `/sbin/vgscan`, ewentualnie z opcjami `--ignorelockingfailure` i/lub `--mknodes`
4. wykonanie `/sbin/vgchange -a y`, ewentualnie z opcją `--ignorelockingfailure`
5. ustawienie odpowiednich uprawnień do urządzeń

W przypadku ręcznej konfiguracji LVM na wczesnych etapach działania systemu może pojawić się problem z tworzeniem pliku blokady - "LVM - Locking type 1 initialisation failed", spowodowane to może być tym iż partycja zawierająca `/var/lock` jest w trybie tylko do odczytu, rozwiązaniem jest np. `mount -t tmpfs tmpfs /var/lock`

Niekiedy (np. niezgodne UUIDy dysków/partycji wchodzących w skład RAIDa) może zajść potrzeba ręcznego wystartowania RAIDa wg podanego trybu i na podanych urządzeniach. W tym celu należy (po zatrzymaniu raida który wystartował z błędem poprzez `mdadm -S /dev/mdX`) skorzystać z opcji `--create --assume-clean` do polecenia mdadm wraz z określeniem jaki typ raida, z ilu dysków i wylistoiwaniem tych dysków, np. `mdadm --create --assume-clean --level=1 --raid-devices=2 /dev/md1 /dev/sda1 /dev/sdb1`. Odczyt UUIDów oraz innych informacji odnośnie RAIDa zapisanych na składowym dysku możliwy jest za pomocą: `mdadm --examine /dev/sda1`. W przypadku kłopotów z mdadm'em przydatne mogą być także opcje `--scan`, `--detail`, `--verbose`.

Zobacz w Sieci: RAID programowy[639], Instalacja Linuksa Na Raid1[640], Migracja Serwera Na RAID1[641], LVM[642], LVM[643].

Konfiguracja sieci bezprzewodowej

W trybie klienta podstawowym narzędziem konfiguracji dostępu sieci bezprzewodowej są narzędzia z pakietu `iwconfig` (umożliwiające włączanie interfejsów, skanowanie, etc) oraz `wpa_supplicant` (umożliwijący dostęp do sieci zabezpieczonych WPA - Przykładowy plik konfiguracyjny).

Możliwe jest także zastosowanie systemu linux (z odpowiednią kartą sieciową) jako access-pointa - służy temu hostapd (Przykładowy plik konfiguracyjny) wraz z serwerem DHCP (i ew. DNS) - np. dnsmasq.

Przykładowa konfiguracja dnsmasq:

```
interface=wlan0
dhcp-range=wlan0,192.168.22.193,192.168.22.222,255.255.255.0,1h
log-dhcp
```

WPA_SUPPLICANT.CONF

```
network={
ssid="my-wifi"

proto=WPA2              # standard: WPA lub WPA2
key_mgmt=WPA-EAP        # sposób zarządzania kluczami głównymi: WPA-EAP, WPA-PSK, ...
pairwise=CCMP           # metoda szyfrowania dla unicast: CCMP (AES), TKIP (RC4), ...
group=CCMP              # metoda szyfrowania dla transmisji grupowej: CCMP (AES), TKIP (RC4),⤸
...
eap=TTLS                # metoda protokołu EAP: TLS (certyfikaty), PEAP (tunelowany w TLS ⤸
EAP), TTLS  (tunelowany w TLS EAP lub PAP/CHAP/MSCHAP/MSCHAPV2)

phase2="auth=PAP"       # metoda protokołu EAP używana wewnątrz tunelu TLS: PAP (Password ⤸
Autentication Protocol), ...
identity="login"        # identyfikator dla PAP:
password="tajne_hasło"  # hasło dla PAP:
}
```

HOSTAPD.CONF

```
#########################################################################
##                      /etc/hostapd/hostapd.conf                      ##
##              z prostym wsparciem dla WPA-EAP / WPA-PSK              ##
##                                                                     ##
## pełen opis pliku w: /usr/share/doc/hostapd/examples/hostapd.conf.gz ##
#########################################################################

## intrefejs, sterownik, itp
interface=wlan0
driver=nl80211
ctrl_interface=/var/run/hostapd
```

```
ctrl_interface_group=0

## SSID, tryb oraz kanał WiFi
ssid=my-wifi
hw_mode=g
channel=10

## szyfrowanie i autoryzacja - ustawienia ogólne
##  - standard: 1 = WPA, 2 = WPA2 (RSN), 3 = WPA/WPA2
wpa=2
##  - sposób zarządzania kluczami głównymi: WPA-PSK lub WPA-EAP
wpa_key_mgmt=WPA-EAP
##  - metoda szyfrowania (transmisji unicast i grupowej) dla WPA i WPA2 (RSN)
wpa_pairwise=CCMP
rsn_pairwise=CCMP

## szyfrowanie i autoryzacja - ustawienia dla WPA-EAP
##  - wsparcie dla autoryzacji IEEE 802.1X (EAP over LAN)
ieee8021x=1
##  - używamy wbudowanego serwera EAP
eap_server=1
##  - lokalizacja pliku z bazą użytkowników
##    prosta postać pliku (logowanie użytkownika user z hasłem password po TTLS/PAP lub ⤸
PEAP/MSCHAPV2):
##      *       TTLS,PEAP
##      "user"  TTLS-PAP,MSCHAPV2       "password"      [2]
##    pełny opis pliku w /usr/share/doc/hostapd/examples/hostapd.eap_user
eap_user_file=/etc/hostapd/eap_users
##  - certyfikat i klucz serwera EAP
ca_cert=/etc/hostapd/ca_cert.pem
server_cert=/etc/hostapd/ca_cert.pem
private_key=/etc/hostapd/ca_key.pem

## szyfrowanie i autoryzacja - ustawienia dla WPA-PSK
##  - ASCII passphrase dla WPA-PSK pre-shared key
# wpa_passphrase=__SECRET__

## inne
beacon_int=100
auth_algs=3
wmm_enabled=1
```

Tworzenie mirroru zainstalowanego systemu

Jako że w systemach POSIXowatych "wszystko jest plikiem", zdecydowaną większość z nich możemy przenosić na inny sprzęt, powielać, backupować po prostu kopiując pliki (wraz z atrybutami). W celu wykonania kopii systemu na innym dysku/partycji (przeniesienia działającego systemu) należy najpierw utworzyć system plików na tym urządzeniu z wykorzystaniem stosownych narzędzi - `fdisk`, `mkfs`, `tune2fs`, następnie podmontować gdzieś ten system plików i przy pomocy `cp -a` skopiować pliki z starego na nowy dysk (warto tutaj skorzystać z symboli wieloznacznych basha np. `/[a-o]*` tak aby wyspecyfikować wszystkie katalogi za wyjątkiem /proc /sys oraz katalogu w którym podmontowaliśmy nowy system). Po skopiowaniu systemu należy odpowiednio zmodyfikować /etc/fstab na nowym systemie oraz konfigurację bootloadera.

Najprostszą metodą stworzenia kopi zapasowej systemu czy też przeniesienia zainstalowanego systemu na kilka maszyn (nie muszą być nawet takie same) jest wykorzystanie programu tar oraz ssh do zdalnego zapisu/odczytu pliku kopi:

```
# utworzenie kopii zapasowej
```

```
cd / && tar -cf - --exclude="./proc/*" --exclude="./sys/*" --exclude="./dev/*""
↪ --exclude="./run/*" --exclude="./mnt/*" --exclude="./srv/* --exclude="./home/*"
↪ --exclude="./tmp/*" . | ssh user@host 'cat > /path/to/file.tar'
# przywrócenie systemu
# uprzednio fdisk, mkfs, mount nowego FS
cd /mount/point/of/root/fs && ssh user@host 'cat /path/to/file.tar' | tar
↪ --numeric-owner -xf -
grub-install --root-directory=/mount/point/of/root/fs --recheck
↪ /dev/device_witch_system
# bezpośrednie kopiowanie pomiędzy dyskami (nowy zamontowany w /mnt):
cd / && tar -cf - --exclude="./proc/*" --exclude="./sys/*" --exclude="./dev/*"
↪ --exclude="./run/*" --exclude="./mnt/*" --exclude="./srv/*" --exclude="./home/*"
↪ --exclude="./tmp/*" . | tar --numeric-owner -C /mnt/ -xf -
```

W przypadku przenoszenia w ten sposób systemu na inną maszynę może być konieczna edycja plików konfiguracyjnych związanych z siecią (dla Debiana `/etc/network/interfaces`, `/etc/udev/rules.d/*-persistent-net.rules`, `/etc/hosts`, `/etc/hostname`, …) oraz w przypadku specyficznych ustawień innych plików …

Uwagi o hasłach

Do zmiany hasła korzystamy z polecenia `passwd` (podając następnie stare hasło i dwukrotnie nowe), zmiana ta nie dotyczy jednak zazwyczaj haseł do innych usług (np.: MySQL'a - hasło to zmieniamy poleceniem `mysqladmin -unazwa-użytkownika -pstare-hasło password nowe-hasło`, uwaga nazwę użytkownika poprzedza u (bez spacji), a stare hasło p (również bez spacji))

Wycinki z Debian Reference

Jest to subiektywny wybór linków do kilku fragmentów Debian Reference[644] (bardziej aktualna wersja[645]).

Podstawy Unix'a:

- Uniksopodobne przetwarzanie tekstów[646], czyli narzędzia do przetwarzania plików tekstowych, wyrażenia regularne itp.
- Uniksopodobny system plików[647], czyli uprawnienia, znaczniki czasu, dowiązania twrde i symboliczne

Warto wiedzieć:

- Harmonogram działań (cron, at)[648]
- Łączenie plików PDF lub PS[649]
- Jak wyłączyć wygaszacz ekranu[650]
- Ustawienie właściwego typu konsoli[651]
- Zmiana konsoli przy pomocy screena[652]. Od razu w tym miejscu nadmienie, iż bardzo przydatną funkcją screena jest możliwośc odpalenia programu w tle i podglądania jego wyjścia, sesje taką tworzymy `screen -m -d komenda` (przez -S możemy określić jej nazwę), następnie możemy się do niej podłączyć `screen -r [pid]` (aby się odłączyć wciskamy Ctrl+a a potem d), aby ustalic pid możemy wylistować działające sesje `screen -ls`. Plik konfiguracyjny screen'a (`~/.screenrc`) pozwala na dodanie do niego np. paska informacyjnego z zegarem itp. bajerami - przykład:
  ```
  backtick 2 0 1    /bin/date +%s
   # podstaw pod %2` wynik podanej komendy, odsiwerzany co 1 sek. i wyswietlany
  ↪ nieskonczona ilosc cykli (0)
  ```

```
caption always "%{+b}$USER@%H %{Gk}%{gw} (load: %l | tytul okna (opcja -t): %t | wynik
↪  komendy: %2`) %{.k} %-21=%{yk}%Y-%m-%d %0c %D"
 # %{} sluza modyfikacji koloru i efektu, %-21 kaze wypisac 21 znakow przed koncem
↪  linii
```

W przypadku używania screena po su na użytkownika innego niż root na ogół dostaniemy "Cannot open your terminal '/dev/pts/*' - please check.", problem jest związany z niezmienianiem właściciela odpowiedniego urządzenia / nie tworzeniem nowego urządzenia przez su. Najprostszą metodą ominięcia tej niedogodności jest odpalenie `script /dev/null`, a dopiero potem `screen` lub `screen -r` (jedynym zauważonym skutkiem ubocznym jest jeden poziom więcej przy wylogowywaniu). Zachęcam także do zapoznania się z skryptem mojego autorstwa generującym pliki konfiguracyjne screena powodujące odpalenie screen'a z wieloma oknami w których wykonane jest wskazana komenda na zdalnych hostach z listy (poprzez ssh) - tytuł okna odpowiada nazwie zdalnego hosta.

— Podstawy testowania sieci[653]
— Usuwanie zawartości plików[654]
— chroot[655]
— Jak weryfikować dowiązania (hardlink)[656]
— Montowanie obrazów systemów plików[657]

MULTISCREEN.SH

```
#!/bin/sh

COMMAND="top"
HOST_LIST="localhost"

FILENAME="`echo $COMMAND | tr ' ' '_'`.screen.conf"
echo "sessionname \"$COMMAND\"" > $FILENAME
echo "caption always \"%{Yk}TOPs: %t   %{rk}Ctrl-a \\\" >> lista okien %{rk}Ctrl-a n >> ⤸
nastepne okno\"" >> $FILENAME

for h in $HOST_LIST; do
 echo "screen -t $h ssh -t $h \"$COMMAND\"" >> $FILENAME
 # warto także zwrócić uwagę na opcję -t w wywołaniu ssh
 # wymusza ona utworzenie wirtualnego terminala
done

screen -c $FILENAME

rm $FILENAME
```

DLA ADMINA:

— Kontrola dostępu poprzez PAM oraz login[658]
— Bezpieczniejsza praca – sudo[659]. sudo jest programem umożliwiającym wykonywanie przez wybranych użytkowników wybranych komend uprzywilejowanych, np. wpis `user1 ALL=PASSWD: ALL, NOPASSWD:/cmd/path` w konfiguracji sudo spowoduje że `user1` po podaniu (swojego hasła) będzie mógł wykonać dowolną komendę z prawami root'a, a do wykonania `/cmd/path` nie będzie musiał podawać hasła (przydatne w skryptach)
— Optymalizacja ustawień sprzętowych[660]
— Alt-SysRq[661], czyli co robić gdy nic już nie działa (zobacz też: Magiczny Klawisz Linuksa SysRq[662])
— Poziomy raportowania na konsolę[663]

Unixowate systemy operacyjne

Wbrew pozorom/wrażeniom które niekiedy można odnieść Linux nie jest jedynym współczesnym ”posixowatym” systemem operacyjnym. Z istotnych pozycji na rynku open-source należy wymienić przynajmniej także rodzinę BSD[664] oraz OpenSolaris[665]'a. Co prawda większość prezentowanych w tym serwisie artykułów powstała w oparciu o system GNU/Linux, a konkretnie dystrybucję Debian, to jednak wiele z tych rozwiązań sprawdzi się w innych dystrybucjach (nie tylko tych debiano-podobnych), a często także innych systemach ”posixowych”.

*BSD

Jest to znana i ceniona rodzina systemów operacyjnych, wywodząca się od oryginalnego Unixa, dystrybuowana na bardzo liberalnej licencji. Do użytkowania i konfiguracji systemów z tej rodziny odnoszę się w kilku artykułach w tym serwisie. Większość informacji dotyczy FreeBSD[666] i OpenBSD[667] co nie znaczy że nie mogą one być przydatne w innych systemach z tej rodziny takich jak NetBSD[668], BSD/OS.

(Open)Solaris

Solaris[669] jest unixowym systemem operacyjnym opracowanym przez Sun jest następcą SunOs'ów i niekiedy nazywany jest SunOS 5.x, gdzie x to numer wersji Solarisa (pierwotnie numery Solarisów poprzedzane były dodatkowo 2.). W wielu aspektach przypomina systemy *BSD, z którymi łączy go wspólny przodek. Elementem wyróżniającym go z pośród innych systemów operacyjnych jest zaawansowany system plików ZFS[670] (Zettabyte File System). System ten posiada wbudowane mechanizmy zarządzania dyskami będące odpowiednikiem programowego RAID[671] oraz LVM[672] (stanowiące dodatkową warstwę abstrakcji pomiędzy sprzętem a samym systemem plików). System obok oficjalnej Sun'owskiej gałęzi posiada także gałąź na wolnej licencji - OpenSolaris[673]. Poniżej przedstawię kilka uwag konfiguracyjnych dotyczących tego systemu, a w szczególności OpenSolarisa.

inne ...

Oczywiście wyżej przedstawione systemy nie wyczerpują całej gamy systemów operacyjnych (nawet jeżeli brać pod uwagę tylko te POSIXowate lub tylko te typu FLOS). Z systemów o których uważam że trzeba tu wspomnieć są jeszcze dwa systemy wywodzące się z architektury PPC (stare Mac'i) - są to Haiku (dawniej OpenBeOS) oraz Darwin.

Haiku[674] jest będącym ciągle w (dość wczesnej) fazie rozwojowej wolnym (na licencji MIT X11) klonem BeOSa. Sam BeOS był posixowym systemem operacyjnym, wyposarzonym w środowisko graficzne, ale bez X-serwera. Oprócz dobrego wsparcia dla multimediów system charakteryzował się interesującym systemem plików z bardzo rozbudowanymi atrybutami (np. wszystkie dane książki adresowej były przechowywane jako atrybuty pustych (‼) plików). W serwisie tym kiedyś było trochę więcej na temat tego systemu - zainteresowanych zapraszam do archiwum oraz działu o C/C++. System pomimo swojej unixowatości odbiegał jednak dość znacznie (np. strukturą katalogów) od powszechnie spotykanych unixów - programy były przechowywane np. w katalogach `/boot/beos/bin` i `/boot/home/config/bin`, ten drugi miał wyższy priorytet w ścieżce wyszukiwania, konfiguracja umieszczona była w `/boot/home/config/settings/`, a za autostart odpowiadał skrypt bashowy `/boot/home/config/boot/UserBootscript`, sam basz natomiast był dostępny tylko jako `/bin/sh` (a nie jako `/bin/bash`).

Darwin[675] jest unixowatym systemem operacyjnym, stanowiącym podstawę dla Mac OS X[676]. Swoimi korzeniami sięga systemów z rodziny BSD.

Jeszcze innym, niekiedy bardzo przydatnym system operacyjnym FLOS jest FreeDOS[677], warta uwagi jest także jego odmiana w formie bootującej dyskietki - Balder[678].

Linki

Zobacz także: `man nazwa-polecenia` i/albo `info nazwa-polecenia` i/albo `nazwa-polecenia --help`.

Zachęcam też do zapoznania się z artykułemi o konfiguracji sieci IP, podstawowych usługach w sieciach komputerowych, bazą przydatnych programów, obróbce multimediów, poradami konfiguracyjnymi.

Zobacz w Sieci: The Debian Administrator's Handbook[679], The Linux Documentation Project - Guides[680], Debian refcard[681], Unix Toolbox[682], Identyfikacja i testowanie sprzętu PC pod Linuksem[683], A Quick Introduction to Unix[684], Guide to Unix[685], Guide to X11[686], FOSS Network Infrastructure and Security[687]

Warty uwagi jest także commandlinefu.com[688] - serwis gromadzący przykłady użycia różnych komend unixowych. Aby wykodnie kożystać z poziomu powłoki możemy w jej pliku startowym zdefiniować funkcję wyświetlającą przykłady dla danego polecenia - np.: `fu() { wget -O - "http://www.commandlinefu.com/commands/matching/$@/$(echo -n $@ | openssl base64)/plaintext" 2>/dev/null; };`

Podstawy sieci IP

Sieć jest to zasadniczo zbiór hostów (urządzeń wykorzystujących sieć do komunikacji, takich jak komputery, drukarki sieciowe, ...) oraz infrastruktury sieciowej (urządzenia aktywne, okablowanie). W przypadku bardziej abstrakcyjnego patrzenia na zagadnienia sieciowe (właśnie od strony protokołów wyższych warstw niż sprzętowa - takich jak IP) na sieć można patrzeć tylko jako na zbiór hostów (posiadających identyfikujące je numery). Dla funkcjonowania sieci konieczne jest zapewnienie dobrze określonych zasad jej działania - właśnie zbiorami takich zasad wymiany informacji są protokoły sieciowe.

W dokumencie tym zajmować się będziemy zbiorem protokołów określanych mianem TCP/IP[689]. Protokoły te operują na pakietach (zasadniczo o pakietach mówimy przy IP[690] i TCP[691] natomiast przy UDP[692] mówimy o datagramach) czyli porcji przesyłanej informacji do której dodane zostały dodatkowe informacje - nagłówek zawierający m.in. adresy odbiorcy i nadawcy. Protokoły te działają w warstwach 3 (sieciowa) i 4 (transportowa) modelu OSI[693] czyli pomiędzy warstwami sprzętowymi (w których funkcjonuje np. Ethernet) a warstwami aplikacji (w których działa np. HTTP[694], SMTP[695], XMPP[696], ...). Właśnie utworzenie zunifikowanych zasad komunikacji w tych warstwach (zwłaszcza w warstwie sieciowej gdzie działa IP) umożliwia istnienie internetu (między sieci), czyli logicznej struktury łączącej różne sprzętowo sieci w jedną sieć logiczną w której mogą swobodnie działać protokoły wyższej warstwy (internet w tym znaczeniu jest innym terminem niż "Internet" (pisany wielką literą), czyli globalna sieć komputerowa).

IP, ICMP, UDP, TCP

Protokół IP (Internet Protocol) odpowiedzialny jest przede wszystkim za sposób adresacji hostów oraz reguły komutacji pakietów (routing). Jest on niejako wspomagany przez kolejny protokół z tej rodziny - ICMP (Internet Control Message Protocol), którego zadaniem jest przekazywanie informacji kontrolnych np. o nieosiągalności hosta docelowego, odrzuceniu przetwarzania pakietu ze względu na zbyt dużą liczbę skoków (gdy wartość pola TTL[697] z nagłówka IP wyniesie zero) a także pingi (zarówno żądanie jak i odpowiedź).

Niejako w oparciu o protokół IP działają protokoły warstwy transportowej takie jak UDP, TCP, czy też (mniej znane protokoły czasu rzeczywistego, transmisji strumieniowych): RTP[698], RTCP[699] i SCTP[700]. Najprostszym protokołem warstwy transmisji wydaje się być UDP, protokół ten umożliwia przesłanie informacji pomiędzy dwoma hostami IP i nie kontroluje on tego czy została ona przesłana poprawnie. Z kolei TCP czuwa nad tym by przesłana informacja dotarła do adresata i była nieuszkodzona, natomiast w przypadku problemów informacja wysyłana jest ponownie. TCP w związku z tym w przeciwieństwie do UDP musi otworzyć połączenie i wykorzystywać je do kontroli poprawności przesłania informacji, wymaga zatem przesłania większej liczby pakietów (co może prowadzić do pewnych opóźnień itp). W związku z tym TCP używany jest tam gdzie konieczna jest kontrola poprawności transmisji (oraz ponowne wysłanie zgubionego pakietu), UDP tam gdzie nie jest to potrzebne (a liczy się czas). Dodatkowo zarówno UDP jak i TCP na każdym z hostów wyróżniają numeryczny identyfikator dla aplikacji/procesu/usługi będącego odbiorcą czy też nadawcą informacji zwany numerem portu (popularne usługi typu WWW, SMTP posiadają określone domyślne porty dla swoich protokołów).

RTP jest dość specyficznym protokołem - często zaliczany jest do warstwy transportowej gdyż działa poniżej typowych protokołów warstwy aplikacji, jednak dane które opisują nie są informacjami systemowymi/sieciowymi a właśnie aplikacyjnymi, ponadto działa on powyżej protokołów transportowych (zazwyczaj nad UDP). Może przenosić w jednej sesji kilka strumieni, ale zazwyczaj dla każdego strumienia tworzona jest osobna sesja (zestaw portów). Protokół ten umożliwia identyfikację zawartości pakietu, identyfikację kolejności pakietów w strumieniu. Pozwala także na przepuszczanie ruchu przez mostki maskujące oryginalne adresy, zmieniające medium, replikujące strumień

(replikowany unicast, który wydaje się być bardzo dobrym rozwiązaniem gdy z jakiś powodów multicast nie może zadziałać), czy też nawet łączące strumienie z kilku źródeł. Współpracujący z nim RTCP służy do przesyłania raportów z informacjami o stratach, opóźnieniach itp oraz wzajemnej synchronizacji mediów pomiędzy strumieniami - np. synchronizacji audio z wideo).

Adresacja i routing

Quality of Service

QoS[701] są to techniki gwarantujące jakość usługi. Najczęściej stosowaną techniką zaliczaną do QoS jest dynamiczny podział przepustowości łącza z wykorzystaniem algorytmów takich jak HTB[702], CBQ[703]. Nie jest to jednak prawdziwy QoS, gdyż mechanizmy te to coś więcej niż tylko prosty podział pasma. O QoS'ie możemy zaczynać mówić gdy system kolejkowania uwzględnia priorytety dla pewnych typów ruchu (m.in. na podstawie pola ToS[704]). Kolejki priorytetowe mogą być realizowane jako wagowe - cyklicznie z każdej klasy wysyłamy proporcjonalnie do wagi, lub bez-wagowy (klasa ważniejsza może zagłodzić klasy o mniejszym priorytecie, gdyż brany jest z ważniejszej dopóki są lub nie pojawią się jeszcze ważniejsze).

Istnieją dwie podstawowe architektury systemów QoS:

- IntServ[705] - Odbiorca w oparciu o informacje uzyskane od nadawcy na temat parametrów transmisji dokonuje rezerwacji zasobów w wszystkich routerach na ścieżce łączącej go z nadawcą (wykorzystuje do tego protokół RSVP[706]), każdy z routerów decyduje czy może spełnić te wymagania (gdy jedne nie może cała operacja nie udaje się). Wyróżnia się klasy ruchu:
  - Best Effort - standardowa
  - Guaranteed Service - gwarancja dostępności pasma i opóźnienia
  - Control Load Service - zapewniamy mało obciążony kanał, ale bez gwarancji

  Jak wspomniałem każdy strumień negocjowany jest indywidualnie, poprzez mechanizm Addmision Control (podawane są specyfikacje filtru - jak identyfikować ten ruch, specyfikacje przepływu, itd). Ponadto informacje o rezerwacji muszą być okresowo odnawiane co skutkuje znacznym ruchem sygnalizacyjnym.
- DiffServ[707] - Nie ma ścisłych gwarancji dla jakiegokolwiek strumienia. Na brzegu użytkownik negocjuje SLA i ruch jest kontrolowany pod względem tego SLA (m.in. ustawianie i kontrola pola ToS). Może występować także zarządca pasma (Bandwidth Broker[708]), z którym klienci komunikują się przy pomocy protokołu RSVP i negocjują z nim kontrakty SLA, natomiast sam BB dokonuje rezerwacji zasobów w routerach i kontroluje ich stan (zazwyczaj z wykorzystaniem protokołu COPS[709]).

Ciekawszym protokołem (ze względu na dawanie gwarancji i globalność - działanie na całej ścieżce) wydaje się IntServ. Niestety ze względu na gorszą skalowalność oraz konieczność zgodności z tym protokołem wszystkich routerów na wybranej ścieżce jest on rzadko spotykany.

Adresy IPv4

Adresy hostów w sieciach TCP/IP nazywane są adresami IP, w wersji 4 protokołu IP jest to 32 bitowa liczba, zapisywana najczęściej w notacji kropkowo dziesiętnej (gdzie każdy bajt (ciąg 8 bitów) zapisywany jest jako liczba dziesiętna rozdzielana kropką od pozostałych). Początkowa (określona przez maskę podsieci[710]) liczba bitów stanowi adres sieci, reszta jest adresem hosta w danej (pod)sieci.

Wspomniana **maska podsieci** jest również 32 bitową liczbą, w której pierwsze N bitów ma wartość jeden a kolejne 0. W wyniku operacji binarnego AND między maską a adresem IP hosta otrzymujemy adres sieci, pozwala ona zatem na ustalenie czy dany adres IP należy do tej samej podsieci co adres IP hosta. Maskę możemy zapisywać w notacji kropkowo-dziesiętnej bądź w formie skróconej gdzie po ukośniku podajemy liczbę jedynek w masce (N).

Maska podsieci informuje także o liczbie numerów IP wchodzących w skład danej sieci (wynosi ona dla IPv6 2). Dwa spośród numerów IP należących do każdej podsieci mają specjalne znaczenie: pierwszy - jest adresem sieci (identyfikuje sieć), ostatni jest adresem rozgłoszeniowym sieci (wysłane na niego informacje trafią do wszystkich hostów w tej sieci). Zachęcam też do zapoznania się z programem demonstrującym zasady adresacji IP: numeracja_ip.c.

Wyróżnia się także specjalne adresy (sieci) IP: 0.0.0.0/0 - cały Internet, 127.0.0.0/8 (głownie adres 127.0.0.1) - pętla zwrotna (czyli komunikacja hosta lokalnego ze sobą), 224.0.0.0/4 - multicast oraz adresy sieci prywatnych: 10.0.0.0 do 10.255.255.255, 172.16.0.0 do 172.31.255.255 i 192.168.0.0 do 192.168.255.255. Warto wspomnieć o tym że kiedyś stosowany był także podział na klasy sieci A, B i C odpowiadające maskom /8, /16 i /24 oraz klasy D, E i F (adresy 224.0.0.0 do 254.0.0.0).

Więcej na temat szóstej wersji IP (IPv6, wprowadzającej m.in. 128 bitowe adresy) tutaj.

Typy transmisji danych - unicast, broadcast i multicast

Z sieciami opartymi na komutacji pakietów (takimi jak IP) najczęściej kojarzona jest transmisja pomiędzy dokładnie dwoma hostami, gdzie każdy pakiet ma jednoznacznie określonego nadawcę, transmisję taką określa się mianem unicast[711]. Oprócz niej możliwe są także transmisje typu broadcast[712] (pakiet wysyłany jest na adres rozgłoszeniowy sieci - wynik operacji binarnego OR między adresem sieci i binarnie zanegowaną maską sieci) - adresatem jest wtedy każdy host w sieci lokalnej oraz transmisje multicast[713] - transmisja do wybranej grupy hostów (tutaj wykorzystywane są adresy należące do sieci 224.0.0.0/4 i identyfikują one "kanał nadawczy" a nie unikalny host. Więcej na temat multicastu tutaj. Ponadto możliwe są transmisje anycast[714] (jest to w zasadzie unicast z inną formą adresacji) - komunikacja w tym wypadku zachodzi pomiędzy dwoma hostami, przy czym host zdalny nie posiada unikalnego w skali globalnej adresu (adres oznacza jakąś grupę hostów lub pojedyncze oddalone hosty rozróżniane na zasadach routingu).

Routing

Internet składa się z szeregu podsieci (wyznaczanych właśnie przez maski), na granicach (w miejscach połączeń) tych sieci pracują routery (urządzenia odpowiedzialne za kierowanie pakietów do odpowiednich sieci). Router jest hostem podłączonym do kilku sieci fizycznych (będących także osobnymi sieciami IP) i skonfigurowany do przekazywania pakietów między nimi. Router na podstawie adresu odbiorcy oraz tablicy routingu (zawierających informacje o tym jakie sieci znajdują się na jakich złączach oraz jakie są kolejne routery do których należy przesyłać pakiet) decyduje jaką trasą ma zostać przesłany dany pakiet. W szczególności najbliższy danemu hostowi router obsługujący przesył pakietów do danej sieci nazywany jest bramką (gateway), w przypadku zwykłych hostów na ogól jest jedna bramka obsługująca siec 0.0.0.0/0 (do której kierowane są wszystkie pakiety nie adresowane do sieci lokalnej). Router wykorzystywany jest również na styku sieci korzystających z prywatnych (nie routowanych) zakresów IP i sieci publicznej, wówczas pełni on także funkcję translacji adresów IP.

Mechanizm Translacji Adresów Sieciowych (NAT[715]) pozwala na udostępnienie jednego połączenia do sieci publicznej (współdzielenie jednego publicznego IP) przez hosty w sieci lokalnej posiadające wyłącznie IP wewnętrzne. Jest to najogólniejsza metoda modyfikacji adresów sieciowych (IP-Masquerade oraz PAT - Port Address Translation są jej szczególnymi przypadkami) pozwala ona na dynamiczne zastępowanie adresów wewnętrznych adresem publicznym, jak również mapowanie kombinacji "IP:port" publicznych na takie kombinacje w sieci wewnętrznej. Pozwala nie tylko na udostępnianie Internetu w sieciach wewnętrznych mających jedno publiczne IP, ale również na rozkładanie obciążenia (np. ruchu do serwera WWW) na kilka hostów w sieci wewnętrznej. Praktyczne realizacje NAT są również najprostszą formą zapory sieciowej zwanej od angielskiego "firewall" ścianą ogniową.

Oczywiście routing nie sprowadza się tylko do statycznie zapisanych reguł na którym z interfejsów jest dojście do danej sieci. Powszechnie wykorzystuje się protokoły routingu[716] dynamicznego (np. BGP[717]). Umożliwiają one automatyczną wymianę informacji o najlepszych trasach pomiędzy routerami, równoważenie obciążenia łącz (pakiet może pójść trasą ogólniej gorszą ale obecnie mniej zajętą), a także redundancję łącz - ta sama pula IP może być rozgłaszana przez kilku różnych dostawców łącz, co powoduje że awaria jednego z nich nie powoduje zaniku dostępności danych hostów.

Więcej szczegółów o podziale łącza i skryptach NAT w skrypcie routera linuksowego. Zobacz w Sieci: Routing dynamiczny @ Wikibooks[718], Typy NATów[719], Sieci nie tak znowu lokalne[720]

Multicast w IPv4

Standardowo mamy dwa typy transmisji: unicast (pakiet zawiera unikalny adres odbiorcy i przeznaczony jest tylko dla niego) oraz broadcast (pakiet wysyłany jest na adres rozgłoszeniowy podsieci i przeznaczony jest dla wszystkich komputerów w tej sieci, pakiety te nie są routowane - nie opuszczają sieci lokalnej, zatem 255.255.255.255, który jakby mogło się wydawać powinien oznaczać każdy komputer w Internecie, oznacza tylko każdy komputer w sieci lokalnej). Jeżeli chcemy myśleć o rozsądnej transmisji klasycznej telewizji przez internet, konieczne wydaje się znalezienie metody transmisji jeden do wielu, niewymagającej powielania identycznych pakietów (różniących się tylko adresem odbiorcy). Taką metoda transmisji jest właśnie multicast.

Multicast korzysta z zarezerwowanej do tego celu klasy adresów[721] "D" - adresy 224.0.0.0 do 239.255.255.255 (adresy 224.0.0.0 do 224.0.0.255 zarezerwowane są na wewnętrzne funkcje protokołu), zakres 224.1.0.0-238.255.255.255

przeznaczony jest na alokowane dynamicznie adresy globalne (wykorzystane do tego mogą być protokoły SDR lub MASC, jednak w chwili obecnej nie został ustalony mechanizm ich alokowania, praktykuje się alokacje sieci /24 dla posiadaczy numerów AS na zasadzie 233.AS.AS.0/24), natomiast 239.0.0.0/8 przeznaczone jest na adresy prywatne wykorzystywane w obrębie swojej "domeny". Komputery nieobsługujące multicastu powinny ignorować te pakiety, natomiast routery nie multicastowe nie przekazywać ich; i zasadniczo tak sie dzieje … . Adresy te mają inne znaczenie niż w przypadku unicastu - nie identyfikują one żadnego konkretnego hosta, wyznaczają natomiast swego rodzaju grupę hostów a właściwie (jako że może to być grupa zmienna w czasie) swego rodzaju odpowiednik kanału telewizyjnego na którym coś jest nadawane (i kto chce ten słucha). Również pole TTL ma tutaj swego rodzaju specjalne znaczenie (z wykorzystaniem tego pola tworzy się regiony - pewne routery brzegowe nie przekazują pakietów o zbyt niskim TTL - pomimo iż jest różny od zera (!), np. poza Warszawę i okolice nie powinien wyjść pakiet o TTL < 32, a poza Europę o TTL < 128).

Router[722] obsługujący Multicast gromadzi informacje od hostów (w tym innych routerów) do których przekazuje ruch o tym jakimi multicastowymi numerami IP są zainteresowane wykorzystując protokół IGMP[723] i przekazują tylko te pakiety. W przypadku najpopularniejszej wersji drugiej tego protokołu router odpytuje co pewien czas czy są jacyś "klienci" zainteresowani odbiorem transmisji multicasowych (pytania wysyłane są na multicastowy adres rozgłoszeniowy - 224.0.0.1 lub gdy są reakcją na zgłoszenie chęci opuszczenia grupy na adres danej grupy). Na pytanie ("Query") odpowiada jeden host (który pierwszy ten lepszy) wysyłając "Report", hosty same z siebie oprócz wspomnianej informacji o opuszczaniu grupy mogą wysłać także "query" gdy chcą rozpocząć odbiór transmisji multicastowej. W wersji trzeciej hosty oprócz adresu grupy mogą poinformować także o źródłach (adresach źródłowych hostów nadających do tej grupy), router również może poinformować o dostępnych źródłach. Pakiety "report" wysyłane są tylko do routerów, ponadto każdy host należący do grupy odpowiada na pakiet "Query". W sieciach opartych na ethernecie do przesyłu pakietów multicastowych wykorzystywane są multicastowe adresy MAC - ich zakres to połowa jednego OUI, a zaczynają się od 0x01005e. Jako że jest ich mniej niż multicatowych adresów IP są one dynamicznie przydzielane grupom multicastowym. Możliwe jest także rozbijanie transmisji multicast na unicastowe przez jakiś serwer w sieci lokalnej (np. dla hostów klienckich nie obsługujących multicastu, bądź w przypadku problemów z NAT).

Wspomniany IGMP działa tylko pomiędzy routerem a hostami. Do wymiany informacji pomiędzy routerami wykorzystuje się inne protokoły routingowe - DVMRP[724], MOSPF[725], PIM-DM[726], PIM-SM[727], … . Routing multicastowy oparty jest zarówno o adres źródła jak i grupy multicastowej, z wymienionych protokołów tylko PIM-SM obsługuje współdzielone drzewa multicastowe, w których routing nie uwzględnia źródła. Ponadto router przekazuje informację tylko gdy pochodzi z interfejsu odpowiedniego dla źródła. **DVMRP** funkcjonuje w oparciu o tworzenie drzew bazujących na adresie sieci źródłowej i grupy multicastowej w sposób całkowicie niezależny od routingu unicastowego, drzewa te są okresowo zalewane ruchem, po czym routery przesyłają informację jakich danych nie chcą (nie mogą poinformować że chcą zacząć odbierać jakąś transmisję). **MOSPF** jest multicastowycm rozszerzeniem OSPF[728], wykorzystuje informacje tamtego protokołu i dodatkowe zapytania o odbiorców multicastowych do tworzenia drzewa w momencie rozpoczęcia nadawania przez jakieś źródło, w przypadku wielu stref zakłada router brzegowy subskrybuje wszystkie grupy jeżeli ma jakichkolwiek odbiorców. Wspomniany już **PIM-SM** funkcjonuje w oparciu Rendezvous Point (RP), do których wysyłane są treści od nadawców (początkowo tunelem unicastowym, a dopiero po odebraniu takich danych RP inicjalizuje ścieżkę multicast do danego źródła). Również do RP kierowane są pakiety "Join" z informacją o chęci zasubskrybowania grupy multicastowej, nie zawierają jednak one informacji o źródle, a ich ścieżka tworzy drzewo multicastowe. Po zestawieniu połączenia poprzez RP następuje przełączenie na najkrótszą ścieżkę (wtedy już cała trasa routingu zawiera informacje o grupie i źródle. Niestety multicast nie działa w obecnym Internecie w skali globalnej gdyż bardzo wiele routerów go nie obsługuje. Nadzieją wydaje się być IPv6 gdzie (w odrobinie odmiennej formie multicast jest od początki wpisany w standard).

Na koniec wspomnę jeszcze o protokołach wyższej warstwy, podstawowym jest odpowiednik UDP (ten zresztą też umożliwia korzystanie z multicastu) - RTP[729] (Real-time Transport Protocol), odpowiednikiem TCP (tutaj nie jest możliwy multicast) jest z kolei Multicast Transport Protocol. RTP służący do przekazywania transmisji gdzie nie jest konieczne potwierdzanie (poprawnego) odbioru - np. transmisji radiowych czy telewizyjnych), zapewnia on natomiast odpowiednia kolejność pakietów. MTP zapewnia dotarcie nienaruszonego pakietu.

Więcej o Multicast: Dźwięk i obraz w Internecie (1): Multicasting[730], RFC 1112

IPv6 - ADRESY, ZMIANY, …

Główną zmianą wprowadzaną przez nową wersję protokołu IP jest wydłużenie adresów do 128 bitów. Zmianie uległ format nagłówka pakietu oraz dodano kilka innych rozszerzeń (np. kompresję i uwierzytelnianie). Tutaj również pewna pula IP została zarezerwowana dla potrzeb transmisji multicast, jednak została ona podzielona na 14 pul odpowiadających różnym zasięgom transmisji. Adresy zapisywane są zazwyczaj w notacji dwukropokowej, polegającej na zapisywaniu 16bitowych części adresu liczbami szesnastkowymi[731] oddzielanymi dwukropkiem, dodatkowo jeden

ciąg zer (o długości będącej wielokrotnością 16 bitów) może być skompresowany (pominięty) co daje w zapisie dwa dwukropki `::`.

Kolejną dość istotną cechą IPv6 jest autokonfiguracja. Polega ona na tym że dla podsieci będących LAN[732]'em przydzielana jest pula z maską /64 co umożliwia tworzenie unikalnych numerów IP w oparciu o (niepowtarzalne) numery sprzętowe MAC; adres taki (dla adresu MAC `11:22:33:44:55:66`) będzie miał postać: `64bitowy_prefiks_sieci:122:33:44:5566` (pierwsza część adresu MAC zwiększana jest o 2, w środku wstawiane jest FFFE). 64 bitowy prefiks sieci jest informacją rozgłaszaną przy pomocy ICMP[733]v6 przez routery (mechanizm radvd); natomiast jeżeli host nie uzyskał wspomnianego prefiksu w jego miejsce wstawiane jest `fe80::` (czyli `fe80:0000:0000:0000`) - taki adres nazywa się "link-local" (nie jest on routowany do sieci zewnętrznych, jednak zawsze (także gdy prefiks został uzyskany) może być używany wewnątrz sieci lokalnej). Radvd rozgłasza także informacje routingowe (takie jak adres bramy - dhcpv6 tego nie potrafi), niestety nie da się rozgłaszać w ten sposób innej od standardowej dla LAN długości prefixu. Oczywiście nadal możemy korzystać z przydziału IP przez DHCP[734] oraz ręcznego przydziału IP. Omawiając wersję 6 IP należy wspomnieć także o zmianach w samej budowie nagłówków IP - przede wszystkim budowa modularna, ale także np. zastąpienie TTL (wyrażanego mniej więcej w sekundach) przez licznik zliczający w dół przekazania pakietu przez routery.

W IPv6 na transmisje multicastowe przeznaczone zostały adresy postaci `ffyx::`, gdzie x odpowiada za zasięg który obejmuje transmisja multicastowa, y koduje natomiast flagi dotyczące typu adresu (0=adres przydzielony na stałe przez IANA - dobrze znane usługi, 1=nie przydzielone stale, ...). Od strony użytkownika najciekwasze wydają się adresy postaci `ff3x::`, które budowane są w oparciu o maksymalnie 64 bitowy prefiks sieci adresów unicast - dla sieci o adresie `prefix::/MM` będzie to adres `ff3x:00mm:prefix:gropid`, gdzie mm jest zapisaną w systemie szesnastkowym wartością MM (RFC3306). W 6 wersji protokołu nie występują adresy rozgłoszeniowe - zostały one zastąpione specyficzną postacią adresów multicastowych (oznaczających wszystkie węzły - `ff0x::1`, wszystkie routery - `ff0x::2`, np. dla x=2 w zakresie Link-Local).

Każdy unicastowy adres IPv6 może pełnić funkcję adresu anycast - pakiety wysyłane na ten adres dostarczane są do najbliższego (wg konfiguracji routingu) hosta, ma to miejsce gdy adres IPv6 jest przypisany do więcej niż jednego interfejsu. W szczególności adresem anycast jest adres sieci postaci `prefix::`. Adresy tego typu w IPv4 służą m.in. w systemie tuneli 6to4 i jest to adres 192.88.99.1.

Głównym problemem dla końcowego użytkownika korporacyjnego, a także dla części ISP w obecnej formie wdrożenia wydaje się być niemożliwość rozgłaszania prefiksów mniejszych od /32. Sytuacja taka ma miejsce pomimo zrezygnowania z hierarchicznego routingu globalnego (RFC3587 mówi o 45 bitowym global routing prefix poprzedzonym 3 bitowym określeniem polityki adresów) i uniemożliwia multihoming oparty o BGP, a także podzielenie klasy /32 posiadanej przez ISP na dwie sieci o różnych politykach routingu (otrzymanie drugiej /32 przed zapełnieniem 80% posiadanego także jest niemożliwe). Najlepszym rozwiązaniem wydawałoby się zgoda na globalne rozgłaszanie BGP prefixów do /48 włącznie (nasuwa się też chęć rozwiązania problemu brakujących AS w następnej wersji BGP poprzez wykorzystanie prefiksu sieci - tak jak zrobiono to w RFC3306 dla multicastu).

Zobacz w Sieci: Omówienie protokołu IPv6[735], Implementacja serwera i klienta DHCPv6 dla systemów rodziny Linux[736], ip46nat - IPv4-IPv6 NAT[737].

Tunele, VPN, szyfrowanie, ...

Sieci IP umożliwiają tworzenie tuneli, polega to na tym że komputery będące końcami takiego tunelu przyjmują z niego dane, rozpakowują je z IP i na przykład wysyłają dalej (w innym standardzie) bądź przetwarzają w jakiś sposób oraz (analogicznie) pakują dane do przesłania w pakiety IP i wysyłają. Od zwykłego przesyłu danych przez IP odróżnia się to tym iż na ogół końce tuneli są konfigurowane dość sztywno (znaczy się jest z góry określane kto z kim) oraz że dość częstą tą drogą przesyłane są inne protokoły sieciowe (również sam IP).

Do najistotniejszych obecnie zastosowań tuneli należą: tunele IPv6 w IPv4 (umożliwiają korzystanie z IPv6 gdy provider go nie dostarcza bezpośrednio), tunele łączące sieci multicastowe oraz Virtual Private Network (VPN) czyli wirtualne sieci prywatne. VPN[738] polega na utworzeniu między siecią lokalną (na ogół w jakiejś firmie) a odległym hostem tunelu dzięki któremu host ten może zachowywać się tak jakby był bezpośrednio włączony do tej sieci (można by powiedzieć że jest mapowany przez serwer obsługujący VPN na host tej sieci). Oczywiście wymaga to aby na takim hoście znajdowało się odpowiednie oprogramowanie przesyłające cały ruch sieciowy (bądź tylko odpowiednią jego część) do drugiej końcówki tunelu oraz odbierało dane stamtąd. Pewnymi problemami dla VPN mogą być NAT[739] po stronie hosta klienckiego (aby mogło działać wywołanie z sieci lokalnej do tego hosta musi być utrzymywane połączenie - inaczej nie jest on widoczny w sieci publicznej przez którą prowadzony jest tunel).

Kolejną kwestią jest kompresja i szyfrowanie transmisji. Jest to powszechnie stosowane przy VPN, jak również przy innych protokołach; kompresję można wykorzystywać nawet w samym HTTP[740], szyfrowanie wykorzystywane jest najczęściej w SSH i pochodnych, HTTPS oraz protokołach odbiorczych poczty. Najczęściej spotykane jest szyfrowanie z kluczami asymetrycznymi typu OpenPGP[741], często w wolnej implementacji GnuPG (GPG). Do najpopularniejszych protokołów należy wspomniany SSH (wraz z pochodnymi takimi jak SCP, SFTP) oraz (wykorzy-

stywany po części przez SSH) działający w warstwie TCP "Secure Sockets Layer" (SSL), na którym opart jest między innymi HTTPS, POP3S, IMAPS, … . Zarówno SSH jak i SSL posiadają wolne implementacje: www.OpenSSH.org[742] i www.OpenSSL.org[743].

Konfiguracja sieci

Polecenia do konfiguracji sieci

- `ip` - konfiguracja sieci IP oraz routingu (umożliwia np. balans obciążenia pomiędzy dwoma łączami - `ip route add default scope global nexthop via $IP1 dev $IF1 weight 1 nexthop via $IP2 dev $IF2 weight 1`)
- `ifconfig` - konfiguracja interfejsów sieciowych, np. konfiguracja numeru IP i wielkości sieci (maska): `ifconfig eth0 up 192.168.24.1 netmask 255.255.0.0` (w FreeBSD `ifconfig bge0 up 192.168.24.1 netmask 255.255.0.0`)
- `dhclient` - konfiguracja automatyczna interfejsu w oparciu o DHCP
- `route` - konfiguracja routingu, np. ustawienie bramki domyślnej: `route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.0.1` (w FreeBSD `route add -net 0.0.0.0 -netmask 0.0.0.0 192.168.0.1`
- `ethtool` - konfiguracja kart ethernetowych pod względem takich parametrów jak prędkość transmisji, duplex, wake on lan, …
- `ifplugstatus` - sprawdzanie czy karta ma link (wykryty kabel)
- `iwconfig` - konfiguracja (większości - do części trzeba użyć `wlanctl-ng`) bezprzewodowych interfejsów sieciowych
- `wpa_supplicant` - konfiguracja większości bezprzewodowych interfejsów sieciowych z wsparciem dla WPA
- `wpa_cli` - konfiguracja większości bezprzewodowych interfejsów sieciowych z wsparciem dla WPA
- `iwlist` - dodatkowe informacje z bezprzewodowych interfejsów sieciowych (przydatna zwłaszcza opcja `scan` pokazująca dostępne sieci
- `arp` - manipulowanie tablicą protokołu ARP
- `brctl` - konfiguracja bridge'ów
  Debian: bridge-utils[744]
- `iptables` - jądrowe reguły filtrowania i routingu pakietów
- `ebtables` - jądrowe reguły filtrowania i routingu pakietów dla bridge'ów
- `arptables` - jądrowe reguły filtrowania i routingu pakietów w warstwie drugiej OSI
- `ping` - podstawowe narzędzie testowania sieci sprawdza czy można połączyć się z danym hostem (w przypadku problemów z DNS, głównie odwrotnym, przydatna jest opcja -n)
  Debian: iputils-ping[745]
- `traceroute` - trasa pakietów do danego hosta
  Debian: traceroute[746] paris-traceroute[747] tcptraceroute[748] traceroute-nanog[749] traceproto[750] iputils-tracepath[751]
- `mtr` - trasa, straty, opóźnienia itp pakietów do danego hosta
  Debian: mtr[752] mtr-tiny[753]
- `nmap` - skaner sieciowy
- `dhcping` - narzędzie do testowania DHCP
- `arping` - narzędzie do pingowania z wykorzystaniem zapytań ARP zamiast ICMP (istnieją dwie zasadnicze odmiany: z iputils oraz z synscan; ta druga zawarta w debianowym pakiecie "arping" umożliwia także pingowanie po adresie MAC (ale nie przez RARP, bo on nie do tego służy), aby to jednak działało host docelowy nie może ignorować pingów rozgłoszeniowych, metoda obejścia opisana jest w README arping'a)
  Debian: arping[754]
- `arp-scan` - wyszukiwanie hostów w oparciu o zapytania ARP (taki arping odpalany w pętli)
- `ndisc6` - narzędzie do testowania ICMPv6 Neighbor Discovery
  Debian: ndisc6[755]
- `rdisc6` - narzędzie do testowania ICMPv6 Router Discovery
  Debian: ndisc6[756]
- `rltraceroute6` - trasa pakietów do danego hosta IPv6 z użyciem UDP/ICMP
  Debian: ndisc6[757]
- `tcptraceroute6` - trasa pakietów do danego hosta IPv6 z użyciem TCP
  Debian: ndisc6[758]
- `tcpspray6` - narzędzie do pomiaru prędkości łącza z użyciem TCP/IP Discard/Echo
  Debian: ndisc6[759]

— na6 - narzędzie do wysyłania pakietów Neighbor Advertisement
  Debian: ipv6toolkit[760]
— ra6 - narzędzie do wysyłania pakietów Router Advertisement
  Debian: ipv6toolkit[761]
— scan6 - narzędzie do skanowania sieci IPv6
  Debian: ipv6toolkit[762]
— nslookup - narzędzia do zamiany adresów dpmenowych na IP i odwrotnie oraz wyciągania innych informacji z DNS (np. rekordy MX)
— dnstracer - śledzenie trasy zapytań DNS
— dnswalk - debuger DNS
— swaks - narzędzie do testowania SMTP
— telnet - programy umożliwiają zdalny dostęp do powłoki, a także (podobnie jak netcat) m.in. testowanie usług sieciowych (takich jak smtp, www, jabber, …)
— netcat6[763] (gnu netcat, netcat) - programiki pozwalające na operowanie wysyłanie pakietów TCP i UDP z definiowaną przez nas zawartością, umożliwia m.in. testowanie usług sieciowych (takich jak smtp, www, jabber, …), więcej w tutaj
  Homepage: http://netcat.sourceforge.net/http://www.vulnwatch.org/netcat/
— ifplugstatus - program informujący o stanie fizycznego połączenia (linku) na karcie ethernetowej
— netstat - informacje o sieci
— iptraf - monitor IP LAN
— tcpdump - monitor sieci, dhcpdump wsparcie dla tcpdump do obsługi DHCP
— dhcpdump - wsparcie dla tcpdump do obsługi DHCP
— wireshark - monitor sieci
  Homepage: http://www.wireshark.org/
— ttcp - testuje prędkość połączenia sieciowego (strona domowa[764], najnowsza wersja[765] oraz mutacja[766])
— iperf - pomiar prędkości połączenia sieciowego
— nload - graficzne (ncurses) pokazywanie wykozystania (prędkości) interfejsów sieciowych

Pliki z konfiguracją sieci

Za konfigurację sieci odpowiada plik /etc/network/interfaces, po dokonaniu w nim zmian należy przeładować je poleceniem /etc/init.d/networking restart i można sprawdzić przez ifconfig :). Wyjątkiem jest konfiguracja DNS (ta używana "normalnie"), która znajduje się w pliku /etc/resolv.conf (jeżeli z jakiś powodów serwery DNS naszego ISP nam nie odpowiadają możemy użyć ogólnodostępnych serwerów OpenDNS[767]).

Plik /etc/network/interfaces (przykład) zawiera konfigurację dla poszczególnych urządzeń sieciowych, sekcja dla danego urządzenia zaczyna się od iface, po czym następuje nazwa urządzenia (np. eth0), typ adresu (np. inet lub inet6 lub …) i rodzaju (np. static, dhcp); jeżeli po nazwie jest dwukropek i liczba (np. eth0:0) oznacza to że jest to alias konfiguracji dla danego urządzenia (odpowiada to ifconfig eth0:0 ...). Wpis ten może być poprzedzony linią auto nazw_interfejsu powodującą automatyczne konfigurowanie interfejsu przy starcie systemu (wywołanie /etc/init.d/networking start), przy braku tego wpisu interfejs należy uruchamiać ręcznie przy pomocy ifconfig nazw_interfejsu up.

Po linii określającej konfigurowane urządzenie występują linie określające jego konfigurację. Dla inet czyli IPv4:

— address - określa adres IP,
— netmask - maskę podsieci,
— broadcast - adres rozgłoszeniowy podsieci (może nie być określany),
— network - adres podsieci (opcjonalne - nie potrzebne dla jąder > 2.0),
— gateway - określa bramkę (opcjonalne),
— dns-nameservers - adres serwera DNS (używany tylko rzez resolvconf).

Możemy także korzystać z post-up pre-down itp. w celu wykonania jakiś poleceń po uruchomieniu interfejsu czy przed jego zakończeniem. W ten sposób możemy konfigurować w tym pliku rzeczy normalnie w nim nie ustawiane:

— konfigurację DNS - post-up echo "nameserver 192.168.0.1" > /etc/resolv.conf
— dodatkowy adres IP - post-up ip address add broadcast + local 192.168.24.101/24 dev eth0, warto wtedy też użyć pre-down ip address del 192.168.24.101/24 dev eth0
— wpisy dodatkowych tras routingu - post-up route add ...

Dla inet6 określa się `address` (w notacji dwukropkowej), `netmask` (w postaci ilości bitów o wartości 1 w masce) oraz (opcjonalny) `gateway`. Przydatne może być też `hwaddress` określające MAC karty sieciowej (zwłaszcza przy niektórych konfiguracjach dhcp).

Z pośród innych ustawień związanych z siecią istotną rolę odgrywa także nazwa hosta lokalnego - ustawiana jest ona w `/etc/hostname` (może być podana w formie "host.domena", a doraźnie może być zmieniana komendą `hostname` oraz nazwa systemu pocztowego ustawiana w `/etc/mailname`. Niektóre z programów wymagają aby nazwa ta wraz z odpowiednim przyporządkowaniem ip znalazła się także w DNSie lub pliku lokalnego rozwijania nazw - `/etc/hosts` (tu też możemy podawać przypisania nazw innych hostów do ich adresów, gdy z jakiś powodów nie chcemy korzystać do tego celu z DNS).

Zobacz też: konfiguracja sieci bezprzewodowej, `man interfaces`, `man resolv.conf`, `man ip`.

INTERFACES

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.0.2
netmask 255.255.0.0

# wpisy network i broadcast można pominąć - wtedy ustawiany automatycznie
broadcast 192.168.255.255
network 192.168.0.0

# trasa domyślna routingu (gdy jest dla danego interfejsu)
gateway 192.168.0.1

# dns ...
post-up echo "nameserver 192.168.0.1" > /etc/resolv.conf

# dodatkowe adresy IP
post-up ip address add broadcast + local 192.168.0.3/24 dev eth0
pre-down ip address del 192.168.0.3/24  dev eth0

# zezwalamy na pingi
post-up echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_all
post-up echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# blokada portu ssh przy częstych próbach logowania
# warto w /etc/ssh/sshd_config ustawić MaxAuthTries 1
post-up iptables -I INPUT -p tcp --dport 22 -m state --state NEW  -m recent --set
post-up iptables -I INPUT -p tcp --dport 22 -m state --state NEW  -m recent --update --
seconds 1800 --hitcount 3 -j DROP
pre-down iptables -F

# dns-* options are implemented by the resolvconf package, if installed
dns-nameservers 192.168.0.1


iface eth0 inet6 static
address 2001:db8::2
netmask 32
gateway 2001:db8::1
# konfiguracja ręczna (wyłączamy automatyczną)
pre-up  echo 0 > /proc/sys/net/ipv6/conf/eth0/autoconf
# zmijszamy MTU 9ważne przy połączeniach tunelowanych)
pre-up  echo 1280 > /proc/sys/net/ipv6/conf/eth0/mtu
```

```
# dodajemy adres zgodnościowy z IPv4
#post-up ip address add local ::192.168.0.1/104 dev eth0
# nie kożystamy z tuneli typu sit
post-up  rmmod sit
# nie kozystamy z adresów link-scope
post-up  for dev in `ip link sh | grep mtu | cut -f2 -d' ' | tr -d ':'`; do ip=`ip addr sh ⤸
dev $dev | grep 'inet6 .* scope link' | awk '{print $2}'`; if [ "$ip" != "" ]; then ip addr ⤸
del $ip dev $dev; fi; done;

auto eth1
iface eth1 inet dhcp
```

Linki

Zachęcam także do zajrzenia do:

- artykułu o konfiguracji routera programowego (zarówno IPv4 jak i IPv6) wraz z przykładami
- artykułu o tunelowaniu SSH
- części "wstępu do techniki" poswięconej telekomunikacji, w której jest także o sieciach IP, ale w trochę innym ujęciu.

Zobacz w Sieci: Linux - podręcznik administratora sieci[768], Sieci lokalne[769], Internet od podstaw[770], Multicast over TCP/IP HOWTO[771], Kształtowanie Ruchu i Zaawansowany Routing HOWTO[772], Zestaw ściągawek na tematy sieciowe[773], Polish IPv6 Task Force[774], DNS Stuff[775], Query the RIPE Whois Database[776], IPv6[777], IPv4[778].

Ethernet

Kable i sprzęt

Kable

Kable zakańczane są gniazdami bądź wtykami typu RJ-45[779] montowanymi według jednego z dwóch schematów kolorystycznych (podana numeracja liczona jest od lewej w gnieździe na które patrzymy od strony frontowej (wtyku), gdzie klips wtyku skierowany jest w dół):

- standard EIA/TIA 568A :
  1. biało-zielony
  2. zielony
  3. biało-pomarańczowy
  4. niebieski
  5. biało-niebieski
  6. pomarańczowy
  7. biało-brązowy
  8. brązowy
- standard EIA/TIA 568B :
  1. biało-pomarańczowy
  2. pomarańczowy
  3. biało-zielony
  4. niebieski
  5. biało-niebieski
  6. zielony
  7. biało-brązowy
  8. brązowy

Kabel który ma oba zakończenia wykonane według tego samego standardu nazywany jest kablem prostym i służy do łączenia hostów z urządzeniami rozdzielającymi sygnał (switche, huby ...). Kabel posiadający jedną stronę wykonaną według standardu A drugą B nazywany jest krosowym (cross) i służy do bezpośredniego łączenia dwóch hostów lub dwóch hubów switchy bez wykorzystania up-link (w zasadzie port oznaczany up-link nie robi nic więcej niż skrosowanie przewodów). Współcześnie coraz większa gama sprzętu sieciowego (switche, ...) posiada system automatycznej detekcji podłączonego typu kabla.

Wtyczki RJ-45 są wtyczkami zaciskanymi na przewodzie (bez konieczności odizolowywania żył). Gniazda RJ-45 najczęściej wykonywane są z złączem typu IDC[780] (Insulation Displacement Connector, KRONE/LSA) służącym do podłączenia przewodu również bez konieczności odizolowywania poszczególnych żył. Gniazda spotykane są najczęściej jako pojedyncze lub podwójne gniazda końcowe oraz panele krosowe zawierające zazwyczaj 16 lub 24 lub 32 itd gniazd. Obie te czynności zazwyczaj wykonuje się specjalistycznymi narzędziami przeznaczonymi do tego celu, ale często też można spotkać się z improwizacją ... .

Wśród skrętki 5 kategorii wyróżnia się: skrętkę nieekranowaną (UTP - Unshielded Twisted Pair, 4 pary przewodów w wspólnej izolacji z tworzywa bez żadnego ekranowania), skrętkę foliowaną (FTP - Foiled Twisted Pair) - wszystkie żyły pokryte wspólną folią skrętkę ekranowaną (STP - Shielded Twisted Pair) - wszystkie żyły pokryte wspólnym oplotem, spotkać można także: FFTP - każda para przewodów jest osobno ekranowana folią, podobnie cały kabel; SFTP - każda para przewodów jest osobno ekranowana folią, cały kabel pokryty jest oplotem. Niekiedy (przy instalacjach telefonicznych, domofonowych itp.) stosuje się skrętki niższej kategorii (najczęściej trzeciej), nie jest to jednak zgodne z modelem okablowania strukturalnego.

Urządzenia sieciowe

Do najważniejszych (aktywnych - nie będących okablowaniem, gniazdami itp.) urządzeń sieciowych zalicza się switch (przełącznik) służący do łączenia segmentów sieci ze sobą. Urządzenie to w sposób "inteligentny" kieruje przekazem informacji w sieciach lokalnych - pamięta on adresy sprzętowe które znajdują się na danym porcie i tylko tam kieruje przeznaczone dla nich informacje. Dawniej funkcję taką pełnił (dzisiaj już w zasadzie nie spotykany) mostek (bridge) będący w zasadzie switchem o dwóch gniazdach, natomiast do łączenia kabli wykorzystywany był (także już nie spotykany) hub (koncentrator) który zapewniał że sygnał nadawany przez dowolny z hostów do niego podłączonych był odbierany przez wszystkie inne (poza nim samym) oraz wzmacniał sygnał. Kolejnymi istotnymi urządzeniami sieciowymi jest transceiver służący do konwersji medium transmisyjnego (głównie między skrętką a światłowodem) oraz regenerator (repeater) wzmacniający sygnał i umożliwiający przekroczenie ograniczenia 100m. Jest jeszcze parę typów urządzeń jednak są one dość rzadko spotykane i mało istotne. Zagadnienie routera - urządzenia łączącego sieci lokalne na bazie protokołu wyższej warstwy, działającego w zasadzie podobnie do switcha, omówione jest w artykule poświęconym IP.

Na koniec dodam tylko że najprostszy hub to w zasadzie drabinka oporników zapewniająca że po pełnym obiegu mamy oporność nieskończonej linii (100omów) - Schemat prostego mini Huba do ethernetu 10Mbit Halfduplex[781], warto zauważyć że można obyć się także bez żadnego huba ... tylko z wykorzystaniem zwykłego trójnika (tutaj komputery wpięte w gniazdka trójnika się nie widzą - widzą tylko ten po drugiej stronie, a w komunikacji dużą rolę odgrywa protokół CSMA/CD) - Rozdzielic LANa[782]

Pakiety i transmisja

Adresy sprzętowe, format ramki i ARP

Jedną z rzeczy która charakteryzuje wszystkie sieci typu Ethernet jest wspólny format ramki (opakowania w które wkładana jest przesyłana porcja informacji, np. datagram UDP) i adresacji, umożliwia to dość łatwe łączenie takich sieci bez korzystania z protokołów wyższej warstwy - takich jak IP. Każde urządzenie które jest nadawcą bądź odbiorcą informacji w sieci Ethernet posiada unikalny 48 bitowy adres MAC (zwany często adresem sprzętowym). Składa się on z dwóch głównych części - w pierwszej zapisany jest numer producenta przydzielany przez IEEE[783] (Institute of Electrical and Electronic Engineers), a w drugiej unikalny numer karty.

Numery MAC identyfikują jednoznacznie hosty w sieci lokalnej, jednak gdy chcemy korzystać z protokołu IP konieczny jest sposób konwersji numerów MAC i numerów IP, zajmuje się tym protokół ARP[784] (zamieniający IP na adresy sprzętowe). ARP wykorzystuje ethernetowski adres rozgłoszeniowy do wysłania zapytania o adres sprzętowy hosta o danym numerze IP, informacje te przechowywane są przez określony czas w tablicy przepisań ARP. Odwrotny w działaniu do ARP (zamieniający MAC na IP) RARP[785] nie działa w oparciu o odpowiedzi hostów tylko w oparciu o odpowiedź serwera tego protokołu i był poprzednikiem rozwiązań typu DHCP.

Dostęp do medium transmisyjnego

Jako że nie ma określonych zasad kto kiedy może nadawać konieczne było stworzenie protokołu umożliwiające wykrywanie zajętości medium transmisyjnego oraz wykrywaniem kolizji (bo na raz może nadawać tylko jeden host w danej domenie kolizji). W sieciach Ethernet zajmuje się tym CSMA/CD[786] (Carrier Sense Multiple Access with Collision Detection - wielodostęp z rozpoznawaniem stanu kanału oraz wykrywaniem kolizji). Przed rozpoczęciem nadawania stacja musi sprawdzić czy medium jest wolne, jeżeli tak może zacząć nadawać, jeżeli dwie stacje zaczną nadawać równocześnie zostaje to wykryte, obie przerywają nadawanie i wznawiają po losowym czasie.

Standard sieci

Najpopularniejszą technologią realizacji sprzętowej sieci lokalnych (LAN - Local Area Network) jest Ethernet[787], aktualnie sieci wykonuje się przeważnie w standardzie 100Base-TX (oznaczenie to rozszyfrowuje się następująco: Base - transmisja w paśmie podstawowym (Baseband Network), T - skrętka (Twisted Pair), X - transmisja po jednej parze w każdą stronę) pozwalającym na transfer 100Mb/s i opartym o topologię "gwiazdy" (star) - kable sieciowe połączone są w jednym wspólnym punkcie, w którym znajduje się koncentrator (hub) lub przełącznik (switch) lub "hierarchicznej gwiazdy" (drzewiastej, tree) - struktura podobna do topologii gwiazdy z tą różnicą, że są tu możliwe gałęzie z wieloma węzłami.

Należy zwrócić także uwagę na zyskujący coraz większą popularność tzw. Gigabit-Ethernet umożliwiający transfer rzędu 1Gb/s i w zależności od wariantu korzystający wyłącznie z kabli miedzianych (w zasadzie powinny być 6 lub 7 kategorii, ale często stosuje się kategorię 5e) bądź także światłowodów.

W omawianym standardzie sieci (100Base-TX) długość kabla pomiędzy dwoma urządzeniami nie powinna przekraczać 100 m, do transmisji wykorzystywane są dwie pary przewodów (nadawcza i odbiorcza) skrętki (kategorii 5 lub 5e (enhanced)), pozostałe przewody można wykorzystać do innych celów jednak nie jest to zalecane. Wykorzystywanie skręconych par przewodów (jeden skręt na 6-10 cm kabla) ma na celu eliminację zakłóceń transmisji - (w uproszczeniu) zakłócenia wchodzą tak samo na oba przewody i różnica miedzy nimi nie zmienia się.

WLAN

WLAN[788] czyli bezprzewodowa sieć lokalna (popularnie określana też mianem WiFi[789], co jest trochę szerszym pojęciem), jest jednym z rodziny standardów około ethernetowych. Korzysta z takiej samej adresacji fizycznej i formatu ramki, jednak w stosunku co do sieci bezprzewodowych musi rozwiązywać dodatkowe problemy związane z ograniczeniem dostępu tylko dla uprawnionych hostów (technologie typu WEP[790], WPA[791]), geograficznym przenikaniem się kilku sieci oraz częstotliwościami na których operuje.

Nowością w stosunku co do kablowego Ethernetu są tryby działania, związane z traktowaniem odebranego sygnału oraz podejściem do autoryzacji. Wyróżnić możemy:

- tryb AP - urządzenie rozsyła sygnał do klientów którzy mogą nawiązać z nim połączenie (w tym dokonać autoryzacji u niego)
- tryb klienta - urządzenie loguje się (autoryzuje się u niego) do urządzenia rozsyłającego sygnał WiFi (Access Point[792]) i odbiera go na swoje potrzeby (standardowa karta wifi w komputerze) bądź na potrzeby rozesłania go kablem (AP client)
- tryb point to point - sygnał jest przesyłany bezprzewodowym tunelem pomiędzy dwoma urządzeniami, urządzenia w tym trybie nie udostępniają "normalnej sieci bezprzewodowej"
- tryb point to multipoint - podobnie jak w "point to point" tylko urządzenie wyższe w hierarchii rozsyła sygnał do kilku urządzeń działających w trybie point to point
- wireless ethernet bridge - w trybie tym połączenia punkt-punkt tworzone są w trybie bezpośrednim (AdHoc), do sieci działającej w takim trybie mogą przyłączać się kolejne urządzenia (także klienci) działające w trybie AdHoc.

Konfiguracja - bridge, VLAN i trunking

W oparciu o komputer z linuxem i kilkoma kartami sieciowymi na pokładzie możemy zrobić nie tylko routing ale także zarządzany switch ("routing" ramek ethernetowskich pomiędzy kartami). Do utworzenia oraz zarządzania mostem składającym się z kilku interfejsów sieciowych (powinny one być zdekonfigutowane od strony IP gdyż od momentu włączenia do bridge'a[793] stają się jego portami) służy komenda `brctl`. Poszczególne porty w bridge'u można wyłączać i włączać komendą `ifconfig ethX down|up`. Sam bridge stanowi natomiast pełnowartościowy

interfejs sieciowy, umożliwiający normalną konfigurację TCP/IP. Domyślnie most uczy się na którym z portów ma jakie MAC adresy (czas przechowywania takiej informacji możemy ustawić przy pomocy `brctl`). Jeżeli chcemy uzyskać prawdziwie zarządzany switch powinniśmy się zainteresować pakietem `ebtables`, który umożliwi filtrowanie ruchu przechodzącego przez bridge (także z wykorzystaniem `iptables` czy `arptables`). Warto zapoznać się z przykładami użycia ebtables[794] oraz opisem współdziałania ebtables i iptables[795] (szczególnie: obrazek[796]). Rozdział drugi tego dokumentu przedstawia ogólne zasady podróżowania pakietów przez reguły ebtables. Następnie mamy omówione różnego rodzaju interakcje z iptables i tak:

- Rysunek 3c przedstawia trasę pakietu adresowanego (MAC) do któregoś z interfejsów bridge'a (a więc zapewne adresowany także ip do urządzenia bridge'a), ponadto adres na który wykonywany jest routing jest także za bridge'em. W ogólności dolna lewa bądź prawa część schematu nie musi być na bridgeu (oznaczanym na niebiesko) może być na zwykłym eth* lub procesie lokalnym (wtedy inaczej wygląda też górna część schematu).
- Rysunek 5 przedstawia trasę pakietu przekazywanego wyłącznie przez bridgea (przechodzi także przez odpowiednie regułki iptables - zależne od ustawień systemu bridge-nf w /proc/sys/net/bridge/), w sytuacji gdy nie korzystamy z physdev
- Rysunek 6a przedstawia tą samą sytuację co rysunek 5, ale gdy wykorzystywane jest physdev (iptables -m physdev –physdev-out ethX), które informujące iptables o źródłowym/docelowym porcie na bridge.
- Rysunek 6d pokazuje przypadek z rysunku 3c gdy ruch wejściowy jest ruchem lokalnym i użyty jest physdev.

Bridge ma niekiedy problemy z włączonymi do niego urządzeniami wifi. Związane jest to z ograniczeniami tego standardu, a przede wszystkim z wymogiem iż urządzenie klienckie (w trybie client) może prezentować tylko jeden MAC adres. Zatem aby mogło ono być elementem bridge'a należy zastosować odpowiednie maskowanie MAC-adresów w ebtables. Innym ominięciem tego problemu w AP[797] jest tryb Wireless Distribution System[798].

Kolejnym aspektem związanym z warstwą ethernetową są VLANy[799], umożliwiające tworzenie wirtualnych sieci w warstwie ethernetowej. Oczywiście możliwa jest przeźroczysta dla komputerów końcowych konfiguracja VLANów (switche wiedzą na którym porcie mają jaki VLAN i pakiety danego VLANu wysyłają tylko do / odbierają tylko z portów które do niego należą. Ma to częste zastosowanie gdy VLANy używane są ze względów bezpieczeństwa. Ciekawym przypadkiem jest jednak gdy komputer otrzymuje znaczniki VLANu i sam może identyfikować z którego VLANu do którego należy jest dany pakiet. Do tworzenia takich VLANów oraz zarządzania nimi służy komenda `vconfig`. Możliwe jest także wykorzystanie interfejsów v-lanowych w bridgowaniu ni uzyskanie w ten sposób switcha z wsparciem dla VLANów. Zobacz w Sieci LiNUX Horizon - Virtual Local Area Network[800].

Inną ciekawą możliwością jest port trunking[801] (bonding) polegający na połączeniu kilku interfejsów w jeden o większej prędkości. Wymaga on załadowania modułu bonding z parametrem mode określającym typ tworzonego połączenia interfejsów. Taki wirtualny interfejs uruchamiany jest przy pomocy `ifconfig`, natomiast do dodawania do niego fizycznych interfejsów służy komenda `ifenslave`. Utworzenie kilku interfejsów z różnym mode jest możliwe poprzez odpowiednie skonfigurowanie powiązań nazw tworzonych interfejsów z modułem i opcjami w konfiguracji modprobe (`/etc/modprobe.d/`). Więcej na jego temat znaleźć możemy na LiNUX Horizon - Bonding (Port Trunking)[802].

Linki

Zobacz w Sieci: Sieci lokalne[803], Linux - podręcznik administratora sieci[804].

O podstawowych usługach sieciowych

konto shell'owe (ssh, scp, sftp, unix)

Jest to usługa polegająca na udostępnieniu kont użytkowników na serwerze (z dostępem zdalnym przez SSH[805], rzadziej telnet) pozwalających na korzystanie z zainstalowanych tam programów oraz innych usług omawianych poniżej, jak również umieszczania tam swoich plików (przy pomocy SCP[806] lub SFTP[807], rzadziej FTP).

SSH (bezpieczny shell) jest protokołem pozwalającym na zdalną pacę w którym połączenie (a w szczególności hasło) jest kodowane, na jego bazie funkcjonują rozwiązania takie jak SCP (bezpieczne kopiowanie) pozwalający na kopiowanie plików między zdalnymi serwerami i SFTP (bezpieczny FTP) pozwalający pracować z komendami FTP przy połączeniu szyfrowanym.

POCZTA (SMTP, POP3, IMAP)

Simple Mail Transfer Protocol[808] to protokół przesyłania poczty elektronicznej pomiędzy agentami dostarczania - MTA (jest to względnie prosty, tekstowy protokół, w którym określa się co najmniej jednego odbiorcę wiadomości (w większości przypadków weryfikowane jest jego istnienie), a następnie przekazuje treść wiadomości). SMTP (także w wersji rozszerzonej m.in. o autoryzację - zobacz SMTP-AUTH[809]) jest często stosowany także do wysyłania poczty z programu pocztowego do pierwszego MTA. SMTP operuje kilkoma typami numerów błędów - są to podstawowe numery zdefiniowane w samym SMTP, numery rozszerzające (rozdzielane kropkami) zdefiniowane w RFC 3463 oraz numery błędów DSN (RFC 1894).

Dopiero odbieranie wiadomości z serwera na którym działa MTA odbywa się przy pomocy innych protokołów takich jak POP3[810] (Post Office Protocol) oraz IMAP (Internet Message Transfer Protocol). IMAP[811] jest protokołem nowszym od POP3 i posiadającym więcej funkcji m.in. obsługę już przefiltrowanej poczty umieszczanej w różnych plikach - skrzynkach (w odróżnieniu od POP3, który jest w stanie pobrać tylko wiadomości ze skrzynki systemowej, pozostawiając je tam lub kasując). Aby odbierać przez IMAP pocztę ze skrzynki systemowej jako `folder` należy podać `INBOX`, natomiast aby odebrać pocztę z innej skrzynki należy podać jej ścieżkę (np. `~/Mail/wazne` odbierze pocztę z pliku ważne umieszczonego w podkatalogu Mail katalogu domowego użytkownika). Warto również zwrócić uwagę na systemy obsługi poczty przez WWW, zarówno przez POP3, IMAP jak i z lokalnych skrzynek, bardzo popularnym programem tego typu obsługującym zasadniczo pocztę z lokalnego serwera jest SquirrelMail. Wart jeszcze wspomnieć o możliwości czytania poczty po zalogowaniu na swoje konto - najczęściej program `mutt` lub poprzez edytor tekstowy.

Systemy obsługi poczty umożliwiają często konfigurację zaawansowanego jej przetwarzania po stronie serwera (takiego jak przenoszenie do odpowiednich folderów pocztowych, autoodpowiadanie, przekazywanie na inne adresy, ...)

NNTP

NNTP[812] - zbliżony do poczty elektronicznej protokół grup dyskusyjnych (wiadomości wysyłane są do grupy o zadanej nazwie, przechowane są na serwerze i mogą zostać pobrane przez osoby czytające daną grupę). Protokół może być wykorzystywany na pojedynczym serwerze lub w ramach sieci serwerów (gdzie te same grupy przechowywane są na wielu serwerach, a użytkownik korzysta z "najbliższego"). W oparciu o ten protokół funkcjonuje sieć usenet[813].

HTTP (HTML, PHP, DB)

Protokół Transmisji Hipertekstu (Hypertext Transfer Protocol[814]) - jest to obecnie jeden z najpopularniejszych protokołów internetowych. Jest on odpowiedzialny za transfer stron WWW[815] (zarówno samego pliku HTML'owskiego jak i elementów w nim umieszczonych - grafiki, dźwięku, ...), obecnie (dzięki możliwości wznawiania przerwanej transmisji) często wykorzystywany również do przesyłu pobieranych z WWW plików. Pozwala również na transfer od użytkownika do serwera (ciasteczka (cookies), formularze, a nawet całe pliki).

W budowie serwisów (oprócz HTML[816]'a, elementów graficznych, skryptów wykonujących się w przeglądarce, ...) często korzysta się z serwerowych języków skryptowych (CGI, PHP, ...) które pozwalają na dynamiczną generację wysyłanej strony (strona sama w sobie zawiera takie same elementy jak strony tworzony w sposób statyczny). Ich wykorzystanie często łączy się z wykorzystaniem systemów baz danych (prawie zawsze są to bazy SQL, do najpopularniejszej należy system MySQL).

DNS

Domain Name System[817] - system pozwalający na posługiwanie się nazwami przyjaznymi dla człowieka zamiast numerami IP (oraz translację odwrotną). Jest jednym z podstawowych protokółów Internetu. DNS oparty jest na hierarchicznym układzie domen (domena nadrzędna to pierwsza od prawej część adresu, natomiast nazwa najbardziej na lewo określa konkretny komputer w danej domenie).

Serwery DNS zawierają różnego rodzaju wpisy, do najważniejszych zalicza się wpisy typu: A (zamiana nazwy na IPv4), AAAA (zamiana nazwy na IPv6), CNAME (alias), MX[818] (serwer obsługi poczty), NS (serwer nazw obsługujący daną poddomenę), SRV[819] (serwer obsługi zadanej usługi), PTR (DNS odwrotny).

Rekordy SRV są bardzo dobrym rozwiązaniem aby móc oferować usługi zlokalizowane na różnych serwerach

pod wspólną nazwą domenową, a także rozkładać obciążenie pomiędzy serwery. Jednak niestety dla wielu (głównie starszych) protokołów implementacja tego mechanizmu praktycznie nie istnieje - jest tak m.in. w wypadku SMTP (tu co prawda mamy MX ale o mniejszych możliwościach niż SRV), HTTP

IRC

IRC[820] - najstarszy, ale nadal bardzo popularny protokół chatów internetowych, rozmowy odbywają się na tzw. kanałach (np. tematycznych), możliwe jest też prowadzenie prywatnej rozmowy z wybraną osobą. Protokół umożliwia tworzenie sieci serwerów przekazujących wiadomości między sobą, wtedy użytkownik komunikuje się z najbliższym siebie serwerem (sieci irc).

Przekaz w czasie rzeczywistym i VoIP

Sieci pakietowe (w szczególności Internet) nie były tworzone z myślą o przesyłaniu danych w czasie rzeczywistym (oczywiście przesyłanie jest stosunkowo szybkie, jednak nie ma gwarancji braku opóźnień ani poprawnej kolejności odbioru pakietów). Jednak popularyzacja tych rozwiązań wraz z chęcią integracji sieci IP z systemami telefonii, radiofonii czy telewizji. O ile w przypadku tych dwóch ostatnich (podobnie jak przy większości usług multimedialnych, np. video na życzenie (VoD[821])) dość prostym rozwiązaniem jest buferowanie po stronie odbiornika (nawet przy transmisjach na żywo kilkudziesięcio-sekundowe opóźnienie jest zaniedbywalne w świetle opóźnień wprowadzanych na etapie realizatorskim ...). O tyle w przypadku telefonii czy wideotelefonii i usług pokrewnych (konferencje, ...) rozwiązanie to nie może być zastosowane a opóźnienia w przesyle informacji muszą być małe (inaczej rozmowa staje się uciążliwa). VoIP[822] (Voice over IP) jest właśnie określeniem zbioru technik i protokołów służących do przesyłania głosu i obrazu w czasie rzeczywistym. Wykorzystuje się m.in. pola pozwalające na priorytetowanie ruchu IP, kompresję danych (często dostosowaną tylko do danego zagadnienia - np. mowy).

transmisje multimedialne - RTSP

RTSP[823] jest protokołem do adresowania URLowego mediów, sterowania odtwarzaniem, cachowaniem strumieni multimedialnych (transmitowanych innym protokołem np, RTP). Zestaw RTSP+RTP dla transmisji obrazu z kamer IP (do celów rejestracji, umieszczanie na www itp). Kamery korzystające z tego zestawu są obsługiwane m.in przez zoneminder[824].

SIP

Session Initiation Protocol (SIP[825]) jest dominującym obecnie standardem tego typu usług (wypiera królujący do niedawna H.323). SIP jest w zasadzie protokołem typu ”peer to peer”, współpracuje z Session Description Protocol (SDP), natomiast do przesyłu swoich sesji wykorzystuje pakiety protokołu RTP[826]. Konstrukcja SIP oparta jest w zasadzie na HTTP[827], podobnie jak on SIP używa prostego mechanizmu żądań i odpowiedzi oraz korzysta z czystego tekstu. Metoda adresacji została natomiast zaczerpnięta z poczty elektronicznej i jest postaci ”uzytkownik@host.domena:port”, SIP korzysta z URI[828] scheme ”sip:” oraz domyślnego portu 5060; możliwa jest tez adresacja w stylu PSTN[829]. Opracowywany jest też protokół IM opary na założeniach SIPu - SIMPLE[830]. Warto wspomnie o serwerach typu registrar oraz proxy - te pierwsze służą wskazywaniu położenia klienta (ale rozmowa nie przechodzi przez ten serwer, chyba że jest do niego jawnie adresowana), te drugie przekazują rozmowę przez siebie (mogą tym samym udostępniać usługi takie jak nagrywanie rozmów itp.). Obok tych serwerów (mogą być niezależnymi usługami) funkcjonuje także nazwa domenowa konta, która w ogólności może być inna od nazw tych serwerów (ale w pewnym sensie musi być rozwijana do adresu serwera registrar.

Zobacz w Sieci: Telefonia @ Wikibooks[831], Bezpieczeństwo VOIP opartego na SIP[832].

XMPP/Jabber

Bardzo interesującym wydaje się protokół XMPP[833]/Jabber[834] (który z dużym prawdopodobieństwem ma szansę stać się oficjalnym protokołem IM). Jest to otwarty, uniwersalny protokół rozproszonej sieci Instant Message (podobnie jak email użytkownik mający konto na serwerze X nie ma problemów z komunikacją z użytkownikiem z serwera Y (są nawet bramki do innych usług IM, takich jak GG, ICQ, …); a nie jak dzieje się w komunikatorach tylko z użytkownikiem swojej sieci).

Protokół ten posiada wiele interesujących rozszerzeń standaryzowanych w ramach dokumentów XEP. Do najciekawszych w mojej opinii należą:

- Jingle[835] (XEP-0166) - czyli natywna (w odróżnieniu od porzuconego już JEP-0111: TINS implementującego sposób nawiązania połączenia SIP) jabberowa komunikacja głosowa i wideo VoIP[836] (w zasadzie to mechanizm jabberowej komunikacji P2P[837]), warto zwrócić uwagę na liczne rozszerzenia takie jak: XEP-0167, XEP-0176, XEP-0177, XEP-0234, XEP-0299, XEP-0266
- Multi-User Chat (XEP-0045) - chat wieloosobowy
- Message Archiving (XEP-0136) - archiwizacja wiadomości po stronie serwera
- Privacy Lists (XEP-0016) - filtrowanie wiadomości po stronie serwera
- Extended Stanza Addressing (XEP-0033) - wieloadresowanie (pola cc, bcc)
- Advanced Message Processing (XEP-0079) - forwardowanie, potwierdzanie dostarczenia, auto-odpowiadanie itp.
- Out-of-Office Messages (XEP-0109)
- Message Forwarding (XEP-0297)
- Moved (XEP-0283)
- Flexible Offline Message Retrieval (XEP-0013) - sprawdzanie wiadomości w trybie off-line
- Stanza Interception and Filtering Technology (XEP-0273)
- Service Delegation (XEP-0291)

TELNET

Telnet[838] jest usługą internetowa pozwalająca na zdalne logowanie się na serwer i zdalną pracę. Hasło przesyłane jest jednak otwartym tekstem co nie jest bezpieczne i dlatego też usługa ta jest bardzo rzadko używana.

Najbardziej przyjemną cechą telnetu (zasadniczo to jego klientów, a nie samego protokołu) jest możliwość tekstowej komunikacji z innymi usługami (działającymi na innych portach i korzystających z innych protokołów), np: WWW, Jabber. Co jest ważną opcją diagnostyczną.

FTP

Protokół Transmisji Plików (File Transfer Protocol[839]) - jest to usługa internetowa pozwalająca na transfer plików z serwera do komputera i z komputera na serwer. Usługa zasadniczo wymaga autoryzacji, często spotykana jest w wariancie dostępu anonimowego uprawniającego tylko do pobierania plików przez ten protokół, podobnie jak w telnecie hasło wysyłane jest jawnym tekstem. Obecnie coraz częściej wypierany przez SFTP/SCP (z dostępu autoryzowanego) i HTTP (z dostępu nieautoryzowanego).

SMB (samba)

Usługa wykorzystywana w sieciach lokalnych, przeznaczona głównie dla komputerów działających pod kontrolą Windows (taki substytut NFS - sieciowego systemu plików znanego z Unixów / Linuxa), pozwalająca na udostępnianie zasobów dyskowych i drukarek w sieci lokalnej.

Inne usługi: finger, identd, daytime, time, …

Usługi te (w dużej mierze obsługiwane przez inetd) obecnie praktycznie nie są już używane (ale np. finger na ogół jest dostępny lokalnie) służyły one (w czasach otwartego internetu) podawaniu informacji o maszynie i jej użytkownikach, a także celom diagnostycznym (echo, chargen), były również przodkami usług obecnie powszechnie

używanych (gopher -> WWW, talk -> IM, rlogin, rsh -> SSH). Wspomniany inetd / xinetd często odpowiada też za usługi dostępu do poczty, telnet i FTP.

Lista ważniejszych programów "komunikacyjnych"

Programy komunikacyjne

Bardzo ważną w dzisiejszych czasach grupą programów są programy odpowiedzialne za komunikację ze światem zewnętrznym.

Należy tu chyba zacząć od przeglądarek internetowych - są to programy umożliwiające przede wszystkim oglądanie stron WWW (dokumentów opisanych w HTML'u i przesyłanych poprzez protokoły HTTP i HTTPS), jednak obecnie pełnią więcej funkcji z których przede wszystkim trzeba wspomnieć o obsłudze protokołu FTP. W śród nich również znajdziemy programy działające w trybie tekstowym (choćby słynny Lynx czy W3M), jak i w trybie graficznym - Mozilla i jej odmiany, Konqueror.

Następną grupą programów komunikacyjnych są klienci poczty elektronicznej oraz grup dyskusyjnych (często ta funkcjonalność jest połączona w jednym programie), należy tutaj wspomnieć o tekstowym pine oraz graficznych klientach. Należy też pamiętać o programach nie będących typowymi klientami ale pomagającym nam w obsłudze poczty takich jak filtr pocztowy (procmail), narzędzie do przetwarzania nagłówków (formail, będący częścią procmail'a) i podbieracz poczty (fechmail).

Kolejną ważną grupą są klienci takich usług jak telnet, ftp, ssh, scp, sftp, …, są to najczęściej programy trybu tekstowego o takich właśnie nazwach, jak również klienci graficznie (głównie usług FTP podobnych).

Wspomnieć należy jeszcze o coraz popularniejszych klientach sieci IM, umożliwiających powiadamianie o obecności w sieci, natychmiastowy przesył wiadomości, przesył plików i rozmowy głosowe (skupie się na otwartej i zdecentralizowanej sieci Jabbera), wśród klientów należy wymienić Psi, Exodus oraz tekstowy imcom (stworzony w Perlu) i centericq (obsługujący również inne sieci IM).

Omawianie programów komunikacyjnych zakończyć należy na klientach p2p (sieci bezpośredniej wymiany plików), klientach protokółu VoIP oraz programach obsługujących fax-modem (faxowanie, telefonowanie, aparat zgłoszeniowy, …) i kartę radiowo-telewizyjną.

- Konqueror[840] - graficzna przeglądarka z KDE
- Mozilla Firefox[841] - graficzna przeglądarka WWW, … (przeglądarka posiada możliwość instalacji licznych rozszerzeń - np. VideoDownloader[842] ułatwiający wyciąganie nagrań wideo z odtwarzaczy flash)
  Debian: iceweasel[843]
- Lynx[844] - tekstowa przeglądarka WWW
- w3m[845] - tekstowa przeglądarka z wsparciem tabelek, obrazków, …
  Debian: w3m[846] w3m-img[847]
- elinks[848] - tekstowa przeglądarka z wsparciem tabelek, obrazków, …
- links2[849] - tekstowa przeglądarka z wsparciem tabelek, obrazków, …
- netsurf[850] - przeglądarka mogąca działać między innymi na framebufferze z wsparciem dla obrazków oraz CSS
  Debian: netsurf-linuxfb[851]
- HTTrack[852] - przeglądarka off-line (tworzenie mirrorów serwisów WWW, przy pobieraniu w sieci LAN czy też z lokalnego komputera warto w nowszych wersjach zwrócić uwagę na przełącznik `-%!` oraz odpowiednio duży parametr zadawany przez `-A`)
  Debian: httrack[853] webhttrack[854]
- wget[855] - pobieranie stron WWW i plików z sieci Web (HTTP, FTP)
- - pobieranie stron WWW i plików z sieci Web (HTTP, FTP)
- kget[856] - pobieracz plików dla KDE
- Claws Mail[857] - graficzny klient e-mail
  Debian: claws-mail[858] claws-mail-feeds-reader[859] claws-mail-address-keeper[860] claws-mail-fancy-plugin[861] claws-mail-trayicon[862] claws-mail-smime-plugin[863] claws-mail-pgpinline[864] claws-mail-pgpmime[865] claws-mail-attach-remover[866]
- Mozilla Thunderbird[867] - graficzna klient poczty …
  Debian: icedove[868]
- Sylpheed[869] - graficzny klient e-mail
- Sylpheed-Claws[870] - graficzny klienty e-mail

- Balsa[871] - graficzny klient e-mail
- Evolution[872] - graficzny klient e-mail
- KMail[873] - graficzny klient e-mail z KDE
  Debian: kmail[874] kmailcvt[875]
- Knode[876] - graficzny klient nntp dla KDE
- Konversation[877] - graficzny klient irc z KDE
- Akregator[878] - graficzny klient RSS z KDE
- Cone[879] - konsolowy klient e-mail
- alpine[880] - konsolowy klient e-mail
- mutt[881] - konsolowy klient e-mail
- mailutils[882] (mailx) - wysyłanie/obsługa poczty elektronicznej z linii poleceń (komenda `mail`)
- sendEmail[883] - wysyłanie e-maili z załącznikami z linii poleceń
  Debian: sendemail[884]
- procmail[885] - przetwarzanie skrzynek pocztowych
- fetchmil[886] - program do odbierania poczty przez POP, IMAP, …
- Psi[887] - klient IM ”Jabber”
- Pidgin[888] - Multikomunikator obsługujący m.in. XMPP (jabber), IRC
- finch[889] - działający w trybie tekstowym odpowiednik Pidgina (m.in. klient xmpp i irc)
- WeeChat[890] - Konsolowy klient xmpp i irc
- mcabber[891] - Konsolowy klient xmpp
- freetalk[892] - Konsolowy klient xmpp
- irssi[893] - Konsolowy klient irc (i nie tylko)
- Egk2[894] - działający w trybie tekstowym komunikator obsługujący m.in. Jabbera
- sendxmpp[895] - narzędzie do wysyłania IM xmpp z linii
- linphone[896] - komunikator VoIP
- ekiga[897] - (gnomemeeting) - komunikator VoIP z obsługą SIP oraz H.323
- kphone[898] - komunikator VoIP z obsługą SIP
- Twinkle[899] - telefon VoIP (SIP)
- I Hear U[900] - telefon VoIP (SIP)
- OpenSSH[901] - ssh, scp, sftp
- VLC media player[902] - sieciowy odtwarzacz (i nadawca) multimediów
- C++ RTMP Server[903] - program do strumieniowania multimediów w sieci
  Debian: crtmpserver[904]

Serwery

Są to programy pracujące w trybie ciągłym i obsługujące nadchodzące (lub zaplanowane) żądania. Do najważniejszych należą serwery usług sieciowych takie jak poczty (sendmail, …), WWW (Apache, …), IMAP, Jabber, FTP, SSH, … oraz wielu usług (xinetd), systemów baz danych (MySQL, PostgreSQL) . Swoistym serwerem jest też systemowy corn realizujący zaplanowane zadania w zadanym czasie. Więcej na ten temat w osobnym artykule poświęconym stawianiu serwera sieciowego.

- Apache[905] - rozbudowany (warto zwrócić uwagę na wiele interesujących modułów) serwer HTTP
- mini-httpd[906] - prosty serwer HTTP
- UW-imap[907] - serwer POP i IMAP
- Dovecot[908] - serwer IMAP i POP
- Exim[909] - serwer SMTP
- Sendmail[910] -
- nullmailer[911] - mały i prosty serwerek odbierający pocztę lokalną (komendy mail itp) i przesyłający ją do wskazanyego serwera SMTP
- esmtp[912] - mały i prosty serwerek SMTP typu relay-only MTA
- msmtp[913] - klient SMTP służący do przekazywania lokalnej poczty do prawdziwego MTA
- - mały i prosty serwerek SMTP służący do przekazywania lokalnej poczty do prawdziwego MTA
- OpenSSH[914] -
- jabberd[915] - serwer sieci IM: Jabber
- jabberd2[916] - serwer sieci IM: Jabber
- ejabberd[917] - serwer sieci IM: Jabber
- MySQL[918] -
- PostgreSQL[919] -

- Xinetd[920] -
- NTP[921] - synchronizacja czasu ...
- Proftpd[922] - rozbudowany serwer FTP (przypominający nieco Apache - zwłaszcza pod względem konfiguracji)
- UDPcast[923] - multicastowy serwer plików
- Flumotion[924] - serwer strumieniowy
- icecast[925] - serwer radia internetowego
- Asterisk[926] - centralka telefoniczna
- DialogPalette[927] - system IVR oparty na PBX Asterisk
- FreeSWITCH[928] - (OpenPBX, fork Asteriska) - serwery SIP, centralki telefoniczne, konkurencyjne wobec Asteriska
- SipX[929] - serwer SIP (konkurencyjny wobec Asteriska)
- CallWeaver[930] - centralki telefoniczne (fork Asteriska)
- phpMyAdmin[931] - webowy interfejs administracyjny MySQL
- phpPgAdmin[932] - webowy interfejs administracyjny PostgreSQL
- phpBB[933] - forum dyskusyjne (warto zwrócić uwagę na wiele interesujących modów)
- SquirrelMail[934] - bramka mailowa (warto zwrócić uwagę na wiele interesujących modów)
- openwebmail[935] - inna bramka mailowa
- jwchat[936] - webowy klienta xmpp/jabber współpracują np. z ejabberd'em
- ijab[937] - webowy klienta xmpp/jabber współpracują np. z ejabberd'em
- Webmnin[938] - administracja przez WWW
- Usermin[939] - zarządzanie kontem (hasło, forward, procmail, ...) przez WWW
- phpWebFileManager[940] - menager plików przez WWW
- dhcp3-server[941] - serwer DHCP
- dibbler-client[942] - klient DHCP6
- radvd[943] - serwer Router Advertisement dla ipv6
- - system obsługi druku
  Debian: lpr[944]
- cups[945] - system obsługi druku

PODSTAWY TESTOWANIA PROTOKOŁÓW SIECIOWYCH - TELNET I NETCAT

Większość podstawowych protokołów sieciowych wykorzystuje w komunikacji wiadomości przesyłane zwykłym tekstem (ASCII[946]), dlatego też do ich testowania możemy wykorzystać klienta telnetu (lub bardziej zaawansowany netcat mogący też pełnić rolę serwera przyjmującego połączenia). Do usług takich należy WWW (protokół HTTP) gdzie po połączeniu się z (standardowo) portem 80 serwera i wysłaniu stosownego komunikatu otrzymamy nagłówki HTTP[947] odpowiedzi oraz jej treść (dane tekstowe np. HTML bądź dane binarne np. obrazek). Wspomniana treść żądania może wyglądać następująco (kończy się wysłaniem pustej linii):

```
GET / HTTP/1.1
Host: www.opcode.eu.org
```

Podobnie jest w protokołach SMTP[948] (przesył poczty elektronicznej) - tutaj możemy korzystając z tej metody wysyłać listy z np. sfałszowanymi nagłówkami, XMPP[949] (jabber), POP3[950], IMAP[951], ... Szczegółowe opisy składni poszczególnych protokołów znaleźć możemy w dokumentach RFC.

LINKI

Warto zapoznać się także z artykułem poświęconym konfiguracji usług sieciowych.

Zobacz w Sieci: URL[952], URN[953], Voice Over Internet Protocol Guru[954], Siproxd[955], Asterisk[956], SIP Express Router[957], Computer network technologies and services[958].

Kilka słów (wstępu) o programowaniu

Generalnie programowanie jest czynnością polegającą na pisaniu programów … czyli chodzi o to aby zmusić komputer do zrobienia czegoś. Wystarczy spojrzeć na hasło Język programowania[959] w Wikipedii aby przekonać się jak bardzo wiele jest różnych języków programowania, jednak wymienienie tych istotnych nie jest większym problemem:

- Asembler[960] (zobacz w Sieci: Art of Assembly Language Programming and HLA by Randall Hyde[961], Asembler @ Wikibooks[962]);
- C i C++, Fortran[963], Ada[964];
- C#[965], Java[966], JavaScript[967] (zobacz skrypt rozwijanego menu i współpracujący arkusz CSS), Object Pascal[968], Objective-C[969], Visual Basic[970];
- AWK, sh,
- Perl, PHP, Python, Ruby, Tcl/Tk;
- za języki programowania uważa się także niekiedy:
  - znacznikowe języki formatowań: SGML[971], XML[972], HTML[973]/XHTML[974] (wraz z arkuszami styli CSS[975]) - zobacz layout beztabelkowy źródła tej strony :-), XML, XSL itp a także zobacz zobacz w Sieci: HTML @ Wikibooks[976], CSS @ Wikibooks[977], SVG @ Wikibooks[978], rozwijane menu na CSS[979], XForms[980]
  - języki skaładu tekstu oraz inne języki specjalistyczne (np. zapytań baz danych): TeX[981] (zobacz artykuł na temat (La)TeX'a), SQL[982] (zobacz w Sieci: Bazy danych @ Wikibooks[983]).

Jednak umiejętność programowania to nie jest znajomość jakiegoś języka, jest to raczej umiejętność logicznego myślenia, myślenia "jak zrobiłby to komputer" i znajomość (dość uniwersalnych) podstaw; dlatego też jeżeli umie się programować w jednym języku to umie się praktycznie w każdym (można bardzo łatwo się nauczyć). Więcej o podstawach programowania i o tym dlaczego warto używać bibliotek, korzystać z istniejących projektów/fragmentów kodu w wstęp do techniki. Zainteresowanym polecam także wykłady dotyczące konstrukcji kompilatorów z strony domowej Jana Pustelnika[984].

Programowanie C/C++ (Linux)

W dziale tym zajmę się programowaniem w C[985] i C++[986] głównie dla Linuxa[987] (ze względu na wykorzystywane funkcje systemowe tego systemu i testowanie w nim prezentowanych rozwiązań). Nie będzie to jednak typowe omówienie czy kurs tego języka, prócz krótkiego wprowadzenia w podstawy tego języka, zamieszczę tutaj szereg bogato komentowanych, niewielkich programów ilustrujących poszczególne zagadnienia. Omówię również korzystanie z ważniejszych/ciekawszych bibliotek (w tym bibliotek standardowych C i C++). Na wstępie zachęcam do zapoznania się z przykładem zamieszczonym w części "wstępu do techniki" poświęconej programowaniu. Zobacz także: C @ Wikibooks[988], C++ @ Wikibooks[989], C & C++ reference[990], cplusplus.com[991].

Podstawy programowania w C

- podstawy.c - podstawy języka - zmienne, instrukcje preprocesora, …
- in_out.c - podstawy języka - standardowe wejście i wyjście, również odczyt i zapis plików w C
- petle.c - podstawy języka - pętle, instrukcje warunkowe, …
- wskaznikologia.c - wskaźniki oraz: funkcje, tablice, przekazywanie do funkcji tablic i nazw innych funkcji, zmienne zależące od zmiennej …
- numeracja_ip.c - prosty program demonstrujący adresację IPv4 oraz operatory bitowe (konwersje napisów na postać numeryczną opisane w programach dotyczących programowania usług sieciowych)

podstawy.c

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje podstawy języka - preprocesor, zmienne itp.
 *
```

```
 */

// komentarzem jest cały tekst od // do końca linii (komentarz C++)
// oraz test zawarty pomiędzy /* i */ (komentarz C)

// na wstepie kilka ogólnych uwag składniowych:
//   * program składa się z ciągu instrukcji, poszczególne instrukcje kończą się ;
//   * bloki instrukcji ujmuje się w klamerki {}
//   * argumenty do funkcji przekazuje się w nawiasach () i rozdziela przecinkami
//         pewnym odstępstwem jest instrukcja for gdzie podajemy 3 bloki instrukcji rozdzielane ⤸
;
//   * ciągi znaków umieszcza się wewnątrz cudzysłowiów "" i są one tablicą char'ów
//   * białe znaki (spacje, tabulatory, nowe linie) na ogół nie odgrywają roli
//         jednym z wyjątków są nawiasy kwadratowe [] po nazwie tablicy w których umieszcza
//         się indeks elementu tablicy - tam nie powinno być spacji


// polecenie preprocesora włącza wskazany plik jeszecze przed kompilacją
// plik może być podany z ścieżką wzgledną wobec bierzącego katalogu np:
// #include "moja_biblioteczka/hello.c"
// może też być podany standardowy plik nagłówkowy
// (scieżka względem katalogu zależnego od systemu - np. /usr/include) np:
#include <stdio.h>
#include <stdlib.h>

// ponizsze polecenie każe zastąpić w kodzie programu KAŻDE wystąpienie _NAPIS_
// poprzez "Hell World"
#define _NAPIS_ "Hello World"
// w podobny sposób możemy definiować pseudo funkcje
#define _SUMA_(a,b) (a + b)
// poprzedzenie argumentu # powoduje ujęcie go w ""
#define _WYPISZ_(a) printf(#a)
// ## powoduje sklejenie ...
// w zasadzie zaleca się ograniczanie #define (w powyższych zasosowaniach)
// na rzecz stałych i funkcji inline

// możemy też warunkować włączanie fragmentów kodu
#define PL

#ifdef PL
 // ten fragment wykona sie tylko gdy zdefiniowana PL
 #undef _NAPIS_
 // oddefiniowalismy _NAPIS_
 #define _NAPIS_ "Witaj World"
 // i zdefiniowaliśmy go inaczej
#endif

// stała napisowa
const char* napis = "Witaj świecie\n";
// funkcja inline - wstawiana literalnie w miejscu wywołania (nie powoduje
// narzutu związanego z normalnymi funkcjami) zwiększa natomiast rozmiar programu
// dostępne w C++ i w współczesnych C (dodane w C99)
inline void wypisz(const char* napis) {
 printf(napis);
}

// zmienne globalna - dostepna w każdej funkcji zdefiniowanej w tym pliku
// uwaga niekiedy może zajść konieczność przypisania wartości dopiero w main()
int kontrolna = 9;
double tablica_kontrolna [2] = {1, 2};
```

```
// deklaracja funkcji, funkcje zdefinujemy sobie po main(), mogłaby też być
// dołączona na etapie linkowania ...
int funkcja (int x, int y);

// funkcja główna programu ... rozpoczęcie programu oznacza wywołania main,
// zakończenie main oznacza zakończenie programu
main(int argc, char *argv[], char *envp[]) {

// zabawy z preprocesorem ...
 printf("%s\n", _NAPIS_);
 printf("%d\n", _SUMA_(5,3));
 _WYPISZ_ (witaj świecie !!! \n);

// wykorzystanie stałych i funkcji inline
 wypisz (napis);

// zmienne globalne
 tablica_kontrolna [1] = 10.5;

// podstawowe operacje matematyczne
 printf("suma: %d\n", 2 + 5);
 printf("różnica: %d\n", 2 - 5);
 printf("mnożenie: %d\n", 2 * 5);
 printf("dzielenie całkowite: %d\n", 2 / 5);
 printf("dzielenie: %f\n", 2.0 / 5.0);
 // uwaga: dzielenie całkowite gdy oba argumenty całkowite
 printf("reszta z dzielenia: %d\n", 2 % 5);

// deklaracja zmiennej
 //    typ nazwa;
 //    typ nazwa2;
 //    typ nazwa = wartosc_poczatkowa;
 //    typ nazwa = wartosc_poczatkowa, nazwa2 = wartosc_poczatkowa2;

// nazwy skladac sie moga z liter, cyfr, znaków _ i zaczynają się od litery

// typy podstawowe - patrz:
 //    http://pl.wikipedia.org/wiki/C_(język_programowania)#Podstawowe_typy
 // warto też zwrócić uwagę na typy stałych np.:
 //    98 to int a 98.0 to double, 98.0F to float,
 //    'c' odpowiada typowi char z wartością równą kodowi ASCI znaku c

// liczby zmienno pozycyjne posiadają wartości oznaczające nieskończoność czy też
 // wynik nie będący liczbą (Not a Number, czyli NaN)
 double a = 0.0/0.0;
 if( !(a==a) )
  printf("a jest NaN\n");
 printf ("%f %f %f\n", 2.0/0.0, -4343.0/0.0, a);
 // uwaga wartości NaN odopiiwada bardzo wiele wartości binarnych zmiennej ...


// mamy też możliwość definicji bardziej zaawansowanych typów:
 int tablica [2];
 // tablica 2 elementowa liczb całkowitych, tablica indeksowana od zera !!!
 // możemy tworzyć aliasy dla typów
 typedef int calkowita;
 calkowita blabla2; // blabla2 jest zmienną int ...
 // jest jeszcze tego trochę ... ale na początek to wystarczy ...
 // patrz też wskaznikologia.c
```

```
// ilosc komorek w tablicy (nie chce dzialac na wskaźnikach)
 printf("Dlugość tablicy: %d\n", sizeof(tablica)/sizeof(tablica[0]) );

{
 double liczba2 = 21.789;
 // poniżej prezentuję rzutowanie czyli konwersje typu, zasad jest taka:
 // (nowy_typ) obiekt_do_konwersji
 int liczba = (int) liczba2;
 printf("%d %f\n", liczba, liczba2);
 }
 //printf("%d %f\n", liczba, liczba2);
 // teraz nie istnieje zmienna liczba2 gdyż zasięg zmiennych ograniczją klamerki

// możemy też konwertować napisy na liczby ...
 // zobacz też man 3 strtod i man 3 atoi
 int liczba = atoi( "43" );
 double liczba2 = atof( "35.71" );
 printf("%d %f\n", liczba, liczba2);

// wywołanie funkcji zwracajacej wartość może wyglądać tak:
 liczba = funkcja (29, 84);
 // jeżeli zwracana wartość nas nie interesuje możemy ją zignorować:
 // funkcja (29, 84);

// na koniec o funkcjach matematycznych - w pliku <math.h>
 // mamy wiele funkcji matematycznych, takich jak: sin, exp, abs - modół, ...
}

// definicja funkcji
int funkcja (int x, int y) {
 int z; // zmienna lokalna funkcji
 z = x - y;
 // funkcja coś robi i zwraca wynik przez return
 // wynikiem może być prosta zmienna (pojedyncza liczba), wskaźnik lub strukturę
 // dzięki tym dwóm ostatnim metodom możemy zwracać z funkcji kilka wartości
 // podobnie jak możemy wykorzystać do tego argumenty przekazywane przez wskaźniki
 return z;
 // mogłoby też być tak: return x - y;
}
```

IN_OUT.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje standardowe wejście i wyjście, równierz odczyt i zapis plików
 *
 */


#include <stdio.h>

main() {
 // najprostsze wypisywanie napisu
 fputs( "Hello World !!!\n", stdout );
 puts( "Hello World !!!\n" );
 // oba wyżej zaprezentowane wywołania są równoważne,
 // przy czym pierwsze można wykorzystać do zapisu do pliku - zamiast stdout
```

```
  // jako drugi argument podajemy identyfikator otwartego do zapisu pliku
  // (o tym za chwilę ...), tak samo jest w wszystkich innych funkcjach gdzie
  // podaje jako agrument stdout lub stdin

// najprostsze wypisywanie napisu
  char c = 'q';
    // definjujemy zmienną całkowitą jednobajtową z przypisaniem
    // numeru ASCII (patrz man ascii) znaku q
  fputc( c, stdout );
  putc( c, stdout ); // starszy wariant ...
  putchar( c );
  putchar( '\n' ); // znak nowej linii wprowadzamy przez \n

// najprostsze czytanie znaku
  puts( "Podaj znak i naciśnij ENTER" );
  int d = fgetc (stdin); // jest to równoważne: int d = getchar();
  getchar();
    // wczytujemy tamten enter
    // ... bo inaczej dalsze czytanie będzie mogło mieć problemy
  if (d != EOF)
    c = (char) d;

// najprostrze czytanie napisu (linii)
  puts( "wpisz coś i naciśnij ENTER" );
  char napis [20]; // 20 elementowa tablica znaków
  fgets( napis, 20, stdin );
  // warto zauważyć że wczytany napis zawiera znak końca linii, jeżeli się
  // zmieścił, a jeżeli nie to dalsze wczytywanie będzie mieć problemy ... istnieje
  // też funkcja gets wymagająca podania tylko nazwy tablicy do której ma czytać

//gets( napis );
  // jednak stosowanie jej jest wyrźnie odradzane:
  // "warning: the 'gets' function is dangerous and should not be used."
  // gdyż niczym nie jest ograniczona ilość wczytywanych znaków,
  // co może prowadzić do błędów przepełnienia bufora ...


// wyjście formatowane
  fprintf(stdout, "%% Podałeś znak %c oraz napis %s\n", c, napis);
    // funkcja ta podstawia w miejsce znaczników %x odpowiednie zmienne
    // - po koleji w miejsce pierwszego %x pierwszy argument po napisie
    // formatującym (u nas za %c zmienna c za %s zmienna napis);
    // znaki wystepujące po % informują jak ma być traktowana dana zmienna,
    // aby uzyskac znak % należy wpisac %% (jak w powyższym przykładzie),
    // więcej o napisach formatujących - man 3 printf
    // uwaga: ilość i typy argumentów muszą się zgdzać !!!
    // a w większości kompilatory tego nie sprawdzają
  sprintf( napis, "Podałeś znak %c", c );
    // tak jk wyżej tyle że do wskazanej tablicy
    // rozszerzania GNU dodają równierz funkcje asprintf, która dostaje
    // wskaźnik, a pamięć potrzebą do stworzenia takiej tablicy
    // przydziela sama ... więcej w napisy.cpp
  printf("napis ma teraz postać: %s\n", napis);
    // tak jak fprintf tyle że zawsze na stdout

// wejście formatowane
  puts( "wpisz liczbę całkowitą\n" );
  fscanf (stdin, "%i", &d);
    // funkcja czyta pierwszy argument z pliku (u nas stdin - czyli
    // standardowego wejścia) i rozkłada go według napisu formatującego,
```

```
    // więcej o napisach formatujących - man 3 scanf
    //
    // funkcję tę można wykorzystać np. do parsowania konfigów "zmienna = wartosc"
    // while ( fscanf(fp, "%10s = %10s\n", name, val) != EOF ) {
    //     if ( strlen(name) == 0 || strlen(val) == 0 ) continue;
    //     ...
    // }
  printf("wczytaliśmy liczbę: %i\n", d);
    // oczywiście jest scanf, które nie ma pierwszego argumentu i domyślnie
    // czyta z sdtin oraz parę innych, w tym sscanf:
  double liczba;
  sscanf("3.6", "%lf", &liczba);
  printf("a to była liczba: %f\n", liczba);
    // uwaga: w *scanf należy dodatkowo pamiętać o tym iż nalezy przekazywać
    // adresy zmiennych do których funkcja ma czytać !!!


// czytanie i pisanie do pliku
  FILE *plik;
  plik=fopen("/tmp/plik1.txt", "w+");
    // otwieramy plik określony w pierwszym argumencie, w trybie określonym
    // w drugim argumencie (r - odczyt, w - zapis, a - dopisywanie,
    // + - dwukierunkowy (używane po r, w albo a)
  fputs("Hello World !!!\n", plik); // wpisujemy do pliku
  printf("aktualna pozycja w pliku to %d\n", ftell(plik));

fflush(plik);
    // jako że są to operacje buforowane to aby mieć pewność że to jest już
    // w pliku należy wykonać fflush(), nie jest to konieczne gdy zamykamy
    // plik (wtedy wykonywane jest z automatu)
  fseek(plik, 0, SEEK_SET);
    // przewijamy do początku jest to równoważne rewind(plik);
  fread(&napis, 20, 1, plik);
    // czytamy z pliku 20 elementów o wielkości 1 bajta do tablicy napis

freopen("/tmp/plik2.txt", "a", plik);
    // zmieniamy nazwę pliku dla naszej struktury FILE
    // oprocz fopen, freopen jest też fdopen, która otwiera plik
    // identyfikowany numerycznym deskryptorem a nie ścieżką i popen,
    // która otwiera potok w wybraną stronę do zadanej komendy
    // przykłady użycia w fork.c
  fwrite(&napis, 20, 1, plik);
    // zapisujemy do pliku 20 elementów o wielkości 1 bajta z tablicy napis
  fclose(plik);
    // zamykamy plik ...

// oczywiście funkcje fwrite i fread można stosować do czytania sdtin i pinania na stdout
  // funkcje te są szczególnie urzytecne przy operacji danymi binarnymi:
  printf("DANE BINARNE:\n");
  fwrite(&liczba, 1, sizeof(double), stdout);
  printf("\nKONIEC DANYCH BINARNYCH:\n");

// mamy też funkcje zmieniające nazwę pliku:
  //    rename("sciezka1", "sciezka2");
  // kasujące plik
  //    remove("sciezka");
  // oraz tworzące unikalną nazwę (nie plik) pliku (uważaną za niebezpieczną): tmpnam()
  // albo zwracającą od razu obiekt typu FILE pliku tymczasowego (usuwanego po
  // zakończeniu programu) - tmpfile(): plik = tmpfile();
  char *nazwa = 0;
```

```
 nazwa = tmpnam( nazwa );
 printf("nasza nazwa: %s\n", nazwa);
}
```

PETLE.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje petle, instrukcje warunkowe itp.
 *
 */


/* podstawowymi instrukcjami wykorzystywanymi w programowaniu są pętle i instrukcje ⤸
warunkowe;
	podobnie jak podtawowe operacje matematyczne zamieniane są one niemalże bezpośrednio na
	instrukcje procesora (w odróżnieniu od wielu innych instrukcji nie są wywołaniami ⤸
	bibliotecznymi
*/

#include <stdio.h>

main() {
 int i = 0;


/// instrukcje iteracyjne (pętle) - powodują wielokrotne wykonywanie tego smego fragmentu ⤸
programu

/// pętla while
 while ( i<4 ) {
  printf("%d", i++);
  // i++; jest (niemalże) równoważne zapisowi i = i+1; czyli zwiekszeniu i o 1
  // możnaby próbować wykonać to też w taki sposób:
  // printf("%d", i=i+1);
  // jednak wtedy program wypisałby 1234 (zamiast 0123), co oznacza iż najpierw
  // i zostało zwiększone o 1 a potem podstawione; jest to równoważne instrukcji:
  // printf("%d", ++i);
  // mamy też podobne operatory zmiejszania o jeden
  //	i-- (podstaw i zmniejsz o jeden) --i (zmniejsz o jeden i podstaw)
 }
 // jak nietrudno zauważyć pętla ta wykonuje następującą po niej instrukcje
 // (bądź blok instrukcji ujęty w nawiasy klamrowe)
 // dopuki warunek podany na początku jest prawdziwy

printf("\n");

/// pętla do
 do {
  printf("%d", i++);
 } while ( i<4 );
 // pętla ta wykonuje instrukcję również dopóki warunek jest prawdziwy,
 // jednak jest on sprawdzany dopiero po wykonaniu bloku instrukcji - zatem nawet pętla ta
 // wykona się zwsze przynajmniej raz (jak widać powyżej),
 // natomiast gdy startujemy z warunku prawdziwego jest równoważna pętli while

printf("\n");
```

```
/// pętla for
 for (i=0; i<4; i++) {
  printf("%d", i);
 }
 // wykonuje się ona bezpośrednio iterując jakiś indeks i jak widać powyższa
 // konstrukcja równoważna jest powyższemu while

printf("\n");


/// instrukcje warunkowe - instrukcje te uzależniają wykonanie fragmętu kodu od zadanego ⤸
warunku,
/// jednak w odróżnieniu od petli fragment ten wykonywany jest tylko raz

/// instrukcja if
 if (i == 4) {
  printf("%d", i++);
 } else {
  printf("!");
 }

if (i == 4) {
  printf("%d", i++);
 } else {
  printf("!");
 }
 // instrukcja ta (printf("%d", i++);) wykona się tylko gdy warunek jest prawdziwy
 // (w tym wypadku i równa się 4), w przeciwnym razie wykona się blok instrukcji
 // zawarty po else (printf("!");), jako że w pierwszym wykonaniu zwiększylismy
 // i to za drugim razem zostanie wypisany wykrzyknik

// oczywiście możliwe są różne inne warianty instrukcji if:
 // bez else:
 // if() {}
 // z wieloma if else
 // if() {} else if() {} else if() {}
 // z wieloma if else i jednym else
 // if() {} else if() {} else if() {} else {}

printf("\n");
 i=1;

/// instrukcja switch
 switch (i) {
  case 0:
   printf("wykonałem dla 0\n");
   break; // przerywamy switch
  case 1:
   printf("wykonałem dla 1\n");
   i = 3;
   // tu nie ma break; więc wykonujemy dalszy ciąg instrukcji
   // (bez względu na wartość i)
  case 2:
  {
   printf("wykonałem dla 2\n");
   break;
  }
  // uwaga jeżeli w ramach swich'a chcemy deklarować zmienne należy ująć
  // całość instrukcji wykonywanych w danym case w klamry ...
  // jak uczyniłem to w tym punkcie ...
```

```c
   case 3:
    printf("wykonałem dla 3\n");
    break;
   default:
    // ta sekcja wykonana byłaby gdyby na wstępie zapodać np. i=5 (zamiast i=1)
    printf("było cos innego niż 0 1 2 3\n");
   //
 }

/* słów parę o warunkach jakie możemy sprawdzać:
 oczywiście oprócz warunku równości (==) możemy też sprawdzać wiele innych rzeczy:
 a<b - a mniejsze od b
 a>b - a większe od b
 a>=b - a nie mniejsze od b
 a<=b - a nie większe od b
 a!=b - a różne od b
 mamy też OR oraz AND logiczne - odpowiedni || oraz &&
   (warunki te sprawdzane są od lewej do prawej i jeżeli wartość logiczna całości
   jest znana (np. prawda OR cos jest zawsze prawdą) sprawdzanie jest przerywane
   - uwaga na wykonywanie funkcji i przypisań pod if, while, itd.)
   możemy też sprawdzać bezpośrednio równość z zerem
 */
if (0) printf("to się nie wypisze\n");
 if (1) printf("a to tak\n");
 if (2) printf("a to ???\n");
 // oczywiście zamist wpisanego na sztywno zera czy jedynki umieszcza się tam zazwyczaj
 // jakąś funkcję zwracająca tekie rawtości ...


/// instrukcje skoków - służą do przemieszczania się w programie ...

/// break
 for (i=0; i<4; i++) {
  printf("%d", i);
  if (i==1) break;
 }
 // wykonanie break; zpowoduje opuszczenie pętli (zawsze tylko najbliższej
 // - najbardziej wewnętrznej pętli w której został użyty)

printf("\n");

/// continue
 for (i=0; i<4; i++) {
  if (i==1) continue;
  printf("%d", i);
 }
 // zpowoduje rozpoczęcie kolejnej iteracji tej pętli
 // (pominiecie instrukcji pomiędzy nim   końcem bloku instrukcji wykonywanych w pętli

printf("\n");

/// goto - powodująca przeskok do określonej etykiety
 /// (powszechnie zaleca się jak najżadsze wykorzystywani tej instrukcji
wypisz_i: // etykieta wypisz_i
 printf("%d", i);

if (i !=0 ) {
  i=0;
  goto wypisz_i; // spowoduje skok do etykiety wypisz_i
 }
```

```
printf("\n");

// return - kończaca działanie funkcji w której została wykonana
  // i zwracająca wartość zadana wartość
  return i;
  // kończymy działanie funkcji main() a więc i całego programu zwracając
  // wartość zmiennej i (u nas będzie to zero - co oznacza że wszystko było OK)
  // to co zwrócił program możemy sprawdzic wpisując w terminalu (na którym działał)
  // po jego wykonaniu: echo $?

// w miejscu tym należy także wspomnieć o możliwości wykonywania tzw. długich skoków
  // polegają one na tym iż z dowolnego miejsca w programie możemy wykonać funkcję
  // longjmp(bufor, kod) powoduje ona przeskok do miejsca w ktorym ostatni raz wywołana była
  // setjmp(bufor) i rozpoczęcie wykonywania programu tak jakby zwróciła ona nierErowa
  // wartość podaną w kod
  //
  // działanie tego mechanizmu polega na tym iż wywołanie setjmp(bufor) powoduje
  // zapamiętanie aktualnego stanu stosu w buforze i zwrócenie przez tą funkcję zera
  // natomiast longjmp(...) powoduje odtworzenie stanu stosu z bufora i zasymulowanie
  // zwrócenia przez setjmp niezerowej wartości
  //
  // nalezy pamiętać iż jest to niskopoziomowy mechanizm, który nie usuwa obiektów
  // dynamicznych tworzonych na stercie, anie nie wywołuje destruktorów
}
```

WSKAZNIKOLOGIA.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody bazujące na wykorzystaniu wskaźników
 *
 * czytanie należy zacząć od funkcji main
 * a w miarę kolejnych wywołań wczytywać się w wcześniej zdeklarowane procedury
 *
 */

#include <stdio.h>

/* ta funkcja nic nie zwraca (void), natomiast odbiera wskaźnik do zmiennej i operuje na ⤸
niej
     w odrożnieniu od funkcji przyjmującej po prostu zmienna x zmiana wartości tej zmiennej
     dokonywana w tej funkcji będzie widoczna po zakończaniu wykonywania tej funkcji */
void funkcja_1 ( int *x ) {
 *x=(*x) + 1; /* pod wartosc pod wskaznikiem x podstawiam watrosc z pod wskaznika x ⤸
 powiększoną o 1 */
}

/* funkcja ta przyjmuje tablice i modyfikuje jej drugi element */
void funkcja_2 ( int tab_x[] ) {
 tab_x[2] = 55;

/* moglibysmy rowniez zwracać (przez return) wskaźnik do tablicy
      nie byłoby natomiast sensu zwracać w funkcji wskaźnika do zmiennej zadeklarowanej w ⤸
      niej
      gdyż po zakończeniu funkcji pamieć w niej rezerwowana jest zwalniana
      i zwracalibyśmy adres do pamieci w której mogą być już zupelnie inne dane */
}
```

```
/* funkcja pomocnicza - dodaje dwie liczby i konweruje zwracana wartosc do double */
double suma (int a, int b) {
 return (double)(a+b);
}

/* zmienna globalna która zostanie zainicjalizowana przed uruchomieniem main (o tym mówi ⤸
static) */
static double mnoznik = 3;
// uwaga: zmienne globalne są alokowane w obszarze pamięci procesu a nie na stosie
//           jak ma to miejsce ze zmiennymi lokalnymi

/* ta funkcja przyjmuje jako argument wskaźnik do funkcji przyjmującej dwa argumenty int i ⤸
zwracającej double */
typedef double (*pomocnicza)(int, int);
double funkcja_3 (double (*f)(int, int)) {
 /* możemy definiować także po prostu wskaźniki do funkcji,
     co więcej możemy do tego wykorzystać definiowane samodzielnie typy */
pomocnicza f2 = f;
 return mnoznik * f(2, 5) + f2(1,1);
}

/* przykładowa strukturka ... */
typedef struct struct_dane dane;
struct struct_dane {
 int liczba;
 char tablica[2];
};

/* funkcja przyjmująca wskaźnik do struktury */
void funkcja_4 (dane *a) {
 /* to jest to samo co: double funkcja_3 (struct struct_dane *a) i wtedy nie potzrzebujemy ⤸
 typedef */
printf( "mamy dwa sposoby odwolania sie do elementów struktury przekaznej przez wskaźnik:")⤸
 ;
 printf( " %d == %d\n", a->liczba, (*a).liczba );
 printf( "i dwa sposoby odwolania sie do adresów elementów tej struktury:");
 printf( " %p == %p\n", &(a->liczba), &((*a).liczba) );
}



/* funkcja startowa przyjmujaca argumenty wywołania programu (tablica argv)
    ich liczbę (argc) a także zmienne środowiskowe (tablica envp)
    oraz zwracającą kod powrotu 0 - sukces, inne - porażka */
int main (int argc, char *argv[], char *envp[]) {

printf("\nWSKAZNIKI\n\n");

int a=5;  /* zmienna całkowita o wartości 5 */
int *wsk=0;  /* wartością wsk jest adres w pamięci ... (w tym wypadku wskazuje na adres ⤸
 zmiennej typu integer),
                    jeżeli void *.... to nie określamy (nie wiemy jeszcze na jakiego typ danych ⤸
                    on wskazuje) */

wsk=&a;
   /* do naszego wskaźnika wpisujemy adres (& oznacza operację pobrania adresu zmiennej) */

printf("Wartosc a wynosi: %d \n", a);
```

```
  /* wypisywanie - zobacz: man 3 printf */
printf("Wartosc pod wskaznikiem: %d \n", *wsk);
  /* * oznacza operator wyluskania, pobiera wartość zmiennej z pod podanego adresu */
printf("Adres zmiennej a %p \n", wsk);

// STAŁOŚĆ WSKAŹNIKÓW
 const int* wskaznik_na_stala=wsk;
 // takiemu wskaźnikowi możemy zmienić zmienną na którą pokazuje,
 // ale nie możemy zmienić wartości tej zmiennej

int* const wskaznik_o_stalym_adresie=wsk;
 // takiemu wskaźnikowi nie możemy zmienić zmiennej na którą pokazuje
 // ale przy jego pomocy możemy zmieniać wskazywaną zmienną

const int* const wskaznik_o_stalym_adresie_na_stala=wsk;
 // temu wskaźnikowi nie możemy zmienić adresu ani przy jego pomocy zmienić wartości ⤸
 wskazywanej


printf("\nTABLICE A WSKAZNIKI\n\n");

/* tablica to w rzeczywistosci wskaźnik do pierwszego jej elementu ... */
int tablica[5]; /* zadeklarowalismy tablicę o pięciu elementach typu int (indeksowanych 0 .⤸
 .. 4) */
tablica[0]=50; tablica[1]=51; tablica[2]=52; tablica[3]=53; tablica[4]=54;

wsk=tablica;
  /* do naszej zmiennej wskaźnikowej wpiszemy adres naszej tablicy (nie potrzebujemy tutaj ⤸
  używać &) */

for (a=0; a<5; a++) {
  printf("Element numer a=%d, wypisany *(tablica+a) wynosi: %d\n", a, *(tablica+a));
  printf("                    i jest to to samo co tablica[a]: %d\n", tablica[a]);
  printf("                    jak równierz wypisanie *(wsk++): %d\n", *(wsk++));
  printf("         jego adres to: %p\n", wsk);
 }


printf("\nFUNKCJE I WSKAZNIKI\n\n");

int z = 0;
 printf("Po zainicjowalizowaniu zmiennej: %d\n", z);

funkcja_1(&z);
 printf("Po wykonaniu funkcji: %d\n", z);

printf("\nFUNKCJE I TABLICE\n\n");

funkcja_2(tablica); /* do tej funkcji przekaże tablicę - kompilator zamieni to na ⤸
 przekazanie wskaźnika */
printf("po naszej kolejnej funkcji wartosc tablica[2] wynosi: %d %d\n", tablica[2], 2[⤸
 tablica]);
 // zaskakujaca postac 2[tablica] jest rownowazna
 // bo operator a[b] jest tak naprawde rozumiany jako *(a+b)


printf("\nFUNKCJE I FUNKCJE\n\n");

double x;
 x = funkcja_3( suma );
```

```
    /* jako argument funkcja_3 przekazujemy nazwe funkcji - suma, a zwracaną wartość ⤸
    wpisujemy do x */
printf("Po wykonaniu naszych funkcji mamy: %f\n", x);


printf("\nTABLICE DWUWYMIAROWE\n\n");

/* deklarujemy tablicę dwuwymiarową i ją wypełniamy */
int tablica_dwuwymiarowa[2][3];
 tablica_dwuwymiarowa[0][0]=0; tablica_dwuwymiarowa[0][1]=1; tablica_dwuwymiarowa[0][2]=2;
 tablica_dwuwymiarowa[1][0]=10; tablica_dwuwymiarowa[1][1]=11; tablica_dwuwymiarowa[1][2]=12;

/* przypisujemy jej adres do naszego wskaźnika
        tym razem użyliśmy * jest to podyktowane tym iz tak naprawde tablica dwuwymiarowa to ⤸
        tablica tablic,
        co zatem idzie pod adresem wskazywanym przez nazwe tablicy jest wskaznik do tablicy ⤸
        odpowiadajcej
        pierwszemu wierszowi (a zarazem pierwszemu elementowi w tym wierszu)*/
wsk = *tablica_dwuwymiarowa;

printf("Wywolanie tablica_dwuwymiarowa[i][j] daje: %d\n", tablica_dwuwymiarowa[1][2]);
 printf("     i to samo daje: wsk[i*SZEROKOSC+j]: %d\n", wsk[1 * 3 + 2]);
    /* taki sposób odwołania przydaje się przy przekazywaniu do funkcji tablic ⤸
    dwuwymiarowych
         gdzie SZEROKOSC to ilość kolumn - zakres drugiego argumentu

    to dziala dlatego ze zazwyczaj i tak calosc takiej tablicy jest alokowan w jednym kawalku,
    ale w ohgolnosci nalezy uwazac bo kolejne wiersze tablicy wielowymiarowej
    nie musza skladac sie na ciagly obszar pamieci ... */


printf("\nTABLICE WSKAZNIKOW i WSKAZNIKI DO WSKAZNIKOW\n\n");

int *wskazniki[2];
 wskazniki[0] = tablica;
 wskazniki[1] = &a;

printf("Element trzeci tablicy siedzacej pod wskazniki[0] to: %d\n", wskazniki[0][3]);
 printf("Ale trzeba UWAZAC bo odwołanie do trzeciego elementu wskazniki[1] nie ma sensu: ⤸
 %d\n", wskazniki[1][3]);
 printf("  sens ma natomiast *wskazniki[1]: %d ale nie wskazniki[1]: %p\n", *wskazniki[1], ⤸
 wskazniki[1]);
 printf("  podobnie *wskazniki[0]: %d i wskazniki[0]: %p\n", *wskazniki[0], wskazniki[0]);
 printf("  natomiast sens posiada **wskazniki (czyli wskaznik do wskaznika, wyluskujacy");
 printf("    zawartosc komurki ktorej adres zapisany jest pod adresem w wskazniki): %d\n", ⤸
 **wskazniki);

/* wskazniki do wskaznikow mozna tez po prostu eklarowac */
int **wskwsk;
 wskwsk = wskazniki + 1;
 printf("Natomiast po podwujnym wyluskaniu wskwsk = wskazniki + 1;, gdzie int **wskwsk; mamy: ⤸
 %d\n", **wskwsk);
 printf("  czyli to samo co *wskazniki[1]\n");


printf("\nSTRUKTURY i WSKAZNIKI DO NICH\n\n");

dane moja_struktura;
 moja_struktura.liczba = 23;
 funkcja_4(&moja_struktura);
```

```
 printf("Co troche moze dziwic ale sama nazwa struktury nie jest traktowana jak wskaznik");
 printf(" lecz jak wyluskanie pierwszego jej elementu: %d\n", moja_struktura);

/* w tym miejscu należy trochę przyszłościowo wspomnieć o wskaźnikach do klas i metod 
 w C++,
   jak już wiadomo wskaźnik zna rozmiar elementu na który wskazuje - dzięki temu działa 
   artmetyka wskaźnikowa,
   ale mechanizm ten może być zdradliwy przy dziedziczonej chierarchii klas - możemy 
   używać wskaźnika na klasę
   bazową celem odwoływania się do klas pochodnych, ale taki wskaźnik ma nieprawdziwą 
   wielkość klasy
   (ma wielkość klasy bazowej a nie pochodnej) pozatym w przypadku dziedziczenia zdradliwe 
   może być
   też porównywanie wartości wskaźników celem sprawdzenia czy wskazują na ten sam obiekt 
   */


return 0;  /* zwracam kod powrotu i zakańczam wykonywanie programu */
}
```

NUMERACJA_IP.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje logiczne operacje bitowe oraz zasady numeracji IP
 *
 */


#include <stdio.h>


void print_binary(unsigned int x) {
 int position = 31;
 int i, y, ctr = 0;

for (i=position; i>=0; i--) {
   y = (x >> i)%2;  // wyciagam kolejny bit w dobrej kolejnosci
   if (ctr == 1)  { // jezeli byl juz niezerowy to ...
    printf("%d", y); // ...    wypisuje
   } else if (y == 1){  // jezeli nie bylo niezerowego a teraz jest to ...
    printf("%d", y); // ... wypisuje i ...
    ctr = 1; // ... zacznaczm ze byl
   }
 }
}
void print_binary_with_zero(unsigned int x) {
 int position = 31; int i;
 for (i=position; i>=0; i--)
  printf("%d", (x >> i)%2);
}


main() {

unsigned int numer_ip = 0xC0A81801; // 192.168.24.1
 unsigned int maska_podsieci = 0xFFFF0000; // 255.255.0.0
```

```
unsigned int adres_sieci = numer_ip & maska_podsieci;
 // logiczny operator bitowy & wykonuje logiczne AND na odpowiadających bitach swoich ↩
 argumentów

unsigned int adres_rozgloszeniowy = numer_ip | ( ~ maska_podsieci);
 // logiczny operator bitowy | wykonuje logiczne OR na odpowiadających bitach swoich ↩
 argumentów
 // logiczny operator bitowy ~ wykonuje negacje każdego bitu swojego argumentu

printf("Numer IP:              "); print_binary_with_zero(numer_ip); printf("\n");
 printf("Maska podsieci:        "); print_binary_with_zero(maska_podsieci); printf("\n");
  // poprawna maska podsieci musi tak wyglądać - ciąg samych jedynek a potem ciąg samych ↩
  zer
 printf("Adres sieci:           "); print_binary_with_zero(adres_sieci); printf("\n");
 printf("Adres rozgłoszeniowy:  "); print_binary_with_zero(adres_rozgloszeniowy); printf(↩
 "\n");

printf("\nAdres sieci to: %X,\n", adres_sieci);
 printf("    co jak łatwo pzeliczyć odpowiada 192.168.0.0:\n        192=0xC0, 168=0xA8\n");
 printf("\nAdres rozgłoszeniowy to: %X,\n", adres_rozgloszeniowy);
 printf("    co jak łatwo pzeliczyć odpowiada 192.168.255.255:\n        192=0xC0, 168=0xA8, ↩
 255=0xff\n");

// aby sprawdzić czy dwa numer należą do tej samej sieci nalezy wykonąć AND binarny z
 // ich maskami podsieci (gdy różne to z maską mniejszej sieci) i porównać wyniki ...


printf("\n\n");


// oprócz poznanych operacji bitowych mamy też EXOR i przesunięcia bitowe
 unsigned int liczba = 0xB9031CAE;
 int liczba2 = 0xB9031CAE;
 printf("jedynka:                         "); print_binary_with_zero(0x1); printf("\n");
 printf("jakaś liczba:                    "); print_binary_with_zero(liczba); printf("\n");
 printf("po operacji EXOR z 0x0000ffff:   "); print_binary_with_zero(liczba ^ 0x0000ffff); ↩
 printf("\n");
 // gdy w drugim argumencie 1 to w bierwszym negujemy ...

printf("\n");

printf("jakaś liczba:                    "); print_binary_with_zero(liczba); printf("\n");
  // liczba bez znaku
 printf("przesunięta o 16 bitów w prawo:  "); print_binary_with_zero(liczba >> 16); printf(↩
 "\n");
  // warto zwrócić uwage że dopełnianie jest zerami ...
 printf("przesunięta o 16 bitów w lewo:   "); print_binary_with_zero(liczba << 16); printf(↩
 "\n");
  // warto zwrócić uwage że dopełnianie jest zerami ...

printf("\n");

printf("jakaś inna liczba:               "); print_binary_with_zero(liczba2); printf("\n");
  // liczba ze znakiem
 printf("przesunięta o 16 bitów w prawo:  "); print_binary_with_zero(liczba2 >> 16); printf(↩
 "\n");
  // warto zwrócić uwage że dopełnianie jest jedynkami ...
 printf("przesunięta o 16 bitów w lewo:   "); print_binary_with_zero(liczba2 << 16); printf(↩
 "\n");
  // warto zwrócić uwage że dopełnianie jest zerami ...
```

```
printf("\n");

// na koniec jak działa funkcja print_binary ...
 printf("jedynka przesunięta o 3 bity w lewo:    "); print_binary(0x1 << 3); printf("\n");

}
```

Podstawy programowania w C++

- in_out.cpp - podstawy języka - standardowe wejście i wyjście, również odczyt i zapis plików w C++, w miejscu tym warto wspomnieć także o bibliotece ułatwiającej generowanie logów log4cxx[992] oraz boost:test ułatwiającym tworzenie testów aplikacji
- klasy.cpp - podstawy programowania obiektowego w C++ (klasy, dziedziczenie, przeciążanie, ...)
- napisy.cpp - napisy języka C oraz C++ - string, sstream, ...
- szablony.cpp - rozwinięcie programowania obiektowego w C++ (szablony, wyjątki, ...)
- stl.cpp - elementy biblioteki standardowej C++ (dynamiczne tablice, listy, algorytmy, ..., zobacz też: Standard Template Library[993])

IN_OUT.CPP

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje standardowe wejście i wyjście, równierz odczyt i zapis plików
 *
 */


// dla C++
#include <iostream>
#include <iomanip> // dla modyfikatorów w cout / cin
#include <fstream> // dla plików
using namespace std; // aby nie pisać std::cout, std::cin, ... itd.
// komentarz: jako że wykorzystywano funkcje biblioteki standardowej C++ przed jej
// ustandaryzowaniem funkcje ustandaryzowane zostały umieszczone w przestrzeni
// nazw std (i korzystają z plików nagłówkowych bez .h)

main() {
 // standardowe wejście, wyjście, wyjście błędu w C++
 double liczba = 0;
 cout << "Podaj liczbę\n"; // wyjście
 cin >> liczba; // wejście - wczytuje pierwszy argument (do białego znaku)
                // i próbuje potraktować jako liczbę,
 cin.clear(); // czyszcze błędy aby program działał normalnie, gdy zostało
              // podane coś co nie może być potraktowane jako liczba
 cerr << "Podałeś " << liczba << "\n"; // wyjście błędu
 clog << "Idziemy dalej\n"; // buforowane wyjście błędu

// odczyt znaku
 int c = cin.get(); // zwraca znak lub EOF
 cout << "Podany został znak o numerze: " << c << " ... podaj kolejne i zakończ Enter" << ⤸
 endl;
 // widać że ze strumienia nie został pobrany Enter, którym zakończylismy wprowadzanie ⤸
 liczby ...
 char d;
 while ( ! cin.get(d).eof() ) { // w tym wypadku EOF sprawdzamy cin.get(d).eof()
```

```
  if (d == 10) break;
  cout << d;
 }
 cout << " - to są podane znaki" << endl;


// odczyt napisu / linii ...
 char napis [20];
 cout << "Wpisz coś (i naciśnij Enter):" << endl;
 cin.getline(napis, 20);
 cout << "wpisałeś: " << napis << endl;

cout << "Wpisz coś (tym razem aby zawierało w środku znak 0):" << endl;
 cin.getline(napis, 20, '0');
 cout << "wpisałeś: " << napis << endl;
 // podobnie można skorzystać z get - cin.get(napis, 20); lub cin.get(napis, 20, '0');
 // wtedy do tablicy wynikowej wpisany zostanie odpowiednio znak nowej linii i 0
 // wczytanie naprawdę cłej linii (bez ograniczenia długości) patrz getline w napisy.cpp

// mamy również odczyt binarny - cin.read( wskaznik_gdzie, ile );

// pozostałe ciekawe metody ...
 //    ignore(_ile_, _ogranicznik_); ignoruje _ile_ znaków albo (co nastapi szybciej)
 //    do _ogranicznik_ domyslnie _ile_=1, _ogranicznik_=EOF
 cin.ignore(1000, '\n'); // ignorujemy wszystko co było dalej

cout << "podaj znak\n";
 cin.get(d);
 cout << "podałeś: " << d << endl;

// mamy też metody:
 //    unget -oddającą ostatnio pobrany znak
 cin.unget();
 //    putback - oddającą zadany znk do strumienia
 // cin.putback(d);
 //    peek - pozwalającą na podejrzeie następngo znaku bez zabierania go ze strumieniac
 cout << "ostatnim pobranym znakiem było: " << cin.peek() << endl;

cin.getline(napis, 20);
 cout << "w naszym strumieniu było: " << napis << endl;


// formatowanie wyjścia/wejścia ...
 cout.setf(ios::fixed);   // zapewnia ze precision() odnosi sie do miejsc po kropce
                          // bez tego precision moze okreslac ilosc znaczacych cyfr w calej ↩
                          liczbie
 cout.setf(ios::showpoint);  // wypisujemy kropkę dziesiętną i nieznaczace zera
 cout.width(10); // minimalna ilosc znakow wypisywanej liczby
 cout.fill('0'); // wypełniacz dla znakow dodawanych na poczatku liczby
 cout.precision(6); // precyzja (patrz uwaga o ios::fixed)
 cout << 20.9863 << endl;

// taki sam efekt możnaby uzyskać korzystając z modyfikatorów umiesczanych w ciągu << <<↩
 :
 cout << setw(6) << setfill('_') << setprecision(2) << 20.9863 << endl;
 // mamy też modyfikatory do zmiany systemu liczbowego
 cout << "10 w systemie ósemkowym: " << setbase(8) << 10 << endl;
 cout << "10 w systemie szesnastkowym: " << setbase(16) << 10 << endl;
 cout << "10 w systemach o podstawie 8, 10 i 16: ";
 cout << oct << 10 << " " << dec << 10 << " " << hex << 10 << endl;
```

```
// warto wspomnieć też o flush wyrzucającym zawartość bufora ...

// są też wspomniane już flagi formatujące ustawiane przez cout.setf(ios::flaga):
 //    skipws - pominięcie początkowych spacji (cin, modyfikator wc)
 //    uppercase - wielkielitery w trybie szesnastkowym i notacji wykładniczej:
 cout.setf(ios::uppercase);
 //    showbase - nagłówki identyfikujące system dla 16-kowego i 8-kowego:
 cout.setf(ios::showbase);
 cout << "10 w systemach o podstawie 8, 10 i 16: ";
 cout << oct << 10 << " " << dec << 10 << " " << hex << 10 << endl;
 //    showpoint - w zmienno przecinkowych zawsze wypisz kropkę
 //    showpos - dodatni poprzedź plusem
 //    unitbuf - wyłącz buferowanie

// ustawiane przez cout.setf(ios::flaga, basefield):
 //    dec - dziesiętny system liczbowy
 //    oct - ósemkowy system liczbowy
 //    hex - szesnastkowy system liczbowy

// ustawiane przez cout.setf(ios::flaga, floatfield):
 //    scientific - notacja naukowa dla zmiennoprzecinkowych
 //    fixed - notacja zwykła dla zmiennoprzecinkowych
 //    0 - automatyczny wybór (domyślna)

// ustawiane przez cout.setf(ios::flaga, adjustfield):
 //    left - wyrównanie do lewej
 //    right - wyrównanie do prawej
 //    internal - wyrównanie do srodka


// obsługa plików
 // oprócz predefiniowanych strumieni możemy tworzyć i korzystac z strumieni zwiazanych z
 plikami
 ofstream plik_w; // tworzymy strumień (typu out) związany z plikiem ...
 ifstream plik_r; // tworzymy strumień (typu in) związany z plikiem ...
 plik_w.open("/tmp/plik_w", ios::out|ios::ate);
  // otwieramy plik w trybie dopisywania z możliwościa zmiany dotychczasowej zawartości
 plik_r.open("/tmp/plik_r"); // otwieramy plik do czytania
 // tryby możemy budować z następujących opcji:
 //    ios::in - odczyt (domyślna dla ifstream)
 //    ios::out - zapis (domyślna dla ofstream)
 //    ios::ate - ustawie pozycji na koniec pliku
 //    ios::app - dopisywanie
 //    ios::trunc - nadpisuje plik
 //    ios::noreplace - nie nadpisuje pliku
 //    ios::nocreate - wymaga aby plik istniał

cout << "Aktualna pozycja w pliku OUT: " << plik_w.tellp() << endl;
 cout << "Aktualna pozycja w pliku IN: " << plik_r.tellg() << endl;

// pisanie i czytanie dokładnie jak dla cin cout ...
 plik_w << "Hello World !!!";

// możemy tez przemieszczać się w pliku - do określonej pozycji
 //    plik_w.seekp( pozycja );
 //    plik_r.seekg( pozycja );
 // lub o określoną odległość (także ujemną) od pozycji
 //    plik_w.seekp( przesuniecie, tryb );
 //    plik_r.seekg( przesuniecie, tryb );
```

```
  // tryb może mieć postaci:
  //    ios::beg - od początku
  //    ios::cur - od bieżącej pozycji
  //    ios::end - od końca

// nma koniec zamykamy pliki
 plik_w.close();
 plik_r.close();
}
```

KLASY.CPP

```
/** plik prezentuje podstawy programowania obiektowego w C++ :
  *   deklaracje klasy
  *   "zasiegowosc zmiennych"
  *   dziedziczenie
  *   deklarowanie konstruktorow i destruktorow
  *   przeciążanie funkcji (na przykładzie konstruktora)
  *   funkcje virtualne
  *
  *   kompilacja: gcc -lstdc++ klasy.cpp
  **/


#include <iostream>
#include <memory>

using namespace std;

/// definicja klasy
class klasa1 {
 private:
  char prywatna; // ta zmienna nie bedzie dostepna z zewnatrz klasy
  // (chyba ze z klasy lub funkcji zadeklarowanej w dej czesci jako zaprzyjazniona:
  //    friend class NAZWA_KLASY )
 public:
  /// konstruktory i destruktor
  klasa1() {
   prywatna = 1;
   publiczna = 6;
  } // odrazu tutaj tez mozna umieszczac cialo funkcji
  klasa1(char t);
  // mozemy definiowac kilka konstruktorow rozniacych sie argumentami
  // (podobnie jak z zwyklymi funkcjami, takze poza klasami - przeciazanie funkcji ...)
  ~klasa1(){
   cout << "to byl destruktor" << endl;
  }

  // klasa zawsze posiada (niezaleznie od tego czy jawnie go zdeklarujemy czy nie):
  //    - konstruktor bezargumentowy
  //    - konstruktor kopiujacy (przyjmujacy obiekt tej klasy jako argument)
  //    - destruktor
  //    - operator przypisania

  /// zmienne
  int publiczna; // tutaj nie mozna przypisac wartosci dlatego robie to w konstruktorach
  static int wspolna; // ta zmienna bedzie wspolna dla wszystkich obiektow tej klasy
  static const int stala = 7; // to bedzie stala,
```

```
    // aby nie tworzyc jej kopi w każdej instancji klasy deklarujemy ją jako static

  /// funkcje
  int funkcja1(int a, int b);
  double funkcja2(double x);
  // możemy deklarować funkcję o tej samej nazwie różniące się typami / liczbą ⤸
  argumentów
  int funkcja2() {
   return 0;
  };
  virtual int funkcja3() {
   return 1;
  };

  // poniższa funkcja4 jest metodą która może być wywołana dla obiektów tej klasy ⤸
  będących "const"
  const int& funkcja4() const {
   cout << "to było dla const" << endl;
   return prywatna;
  };
  // a to jej odpowiednik dla obiektów nie będących const
  int& funkcja4() {
   cout << "to było nie dla const" << endl;

   // zamiast przepisywać kod możemy tutaj wywołać metodą dla obiektów const
   return const_cast<int&>( static_cast<const klasa1*>(this)->funkcja4() );

   // przy okazji warto wspomnieć o operatorach rzutowania w stylu C++
   // w C++ mamy 4 operatory rzutowania:
   //    - static_cast - pomiedzy typami spokrewnionymi
   //    - dynamic_cast - w dół hierarchii klas
   //    - const_cast - zniesienie const (chyba jedyny racjonalny przypadek pokazany wyżej)
   //    - reinterpreter_cast - dowolna konwersja
   //         (dzięki niemu możemy tak jak w C rzutoować dowolny obiekt na dowolny)
   // używa się ich operator<typ_na_kory_zutujemy>(obiekt_ktory_rzutujemy)
  };
};

/// deklaracja zmiennej statycznej z nadaniem jej wartości - jest to niezbeędne aby byla ⤸
ona widoczna ...
int klasa1::wspolna = 0;

/// deklaracja funkcji znajdujących się w naszej klasie (metod)
/// jest to funkcja z wartosciami domyslnymi (coś takiego nie przejdzie w C !!!)
int klasa1::funkcja1(int a = 0, int b = 0) {
 return a + b;
}
double klasa1::funkcja2(double x) {
 return x/prywatna;
}
// konstruktor
klasa1::klasa1(char t) {
 prywatna = t;
 publiczna = 6;
}



/// definicja klasy potomnej dla klasa1 (dziedziczacej z klasa1)
///
```

```
/// klasa ta dziedziczy publicznie czyli publiczne metody klasa1 będą publicznymi w klasa2
/// możliwe jest dziedziczenie private i protected (dostępne tylko dla dziedziczących)
///
/// poza tym warto wspomnieć o przydatnym gdy chcemy korzystać z dziedziczenia ⤸
wielobazowego
/// po kilku klasach mających wspólnego przodka dziedziczeniu wirtualnym (gdy tak będziemy
/// definiowali dziedziczenie po wspólnym przodku to w klasie dziedziczącej wielokrotnie
/// po jego potomkach będzie on włączony jednokrotnie)
class klasa2 : public klasa1 {
 public:
  static int funkcja1(int a = 1, int b = 1) {
   return a * b;
  }
  virtual int funkcja3() {
   return 2;
  }
  int funkcja2() {
   return 3;
  }
  // aby nie nadpisało to "przeciążonych" funkcji o tej nazwie należy dać
  using klasa1::funkcja2;

  klasa2(char t);
  klasa2();

  // deklarowanie funktora
  void operator()(int i, int j) {
   printf("%d\n", i + j);
  }
};

/// dziedziczymy konstruktory
klasa2::klasa2(char t):klasa1(t) {}
klasa2::klasa2():klasa1() {}


/// przeciążenie operatora
int operator+ (klasa2 & kl, int liczba) {
 // tutaj argumenty pobrane zostały przez referncje ...
 // dziala to podobnie do wskaznikow przy czym adres przypisujemy tylko raz
 // a korzystamy bez wyłuskiwania wartości (jak z zmiennej a nie wskaznika)
 // operujemy jednak na orginalnej zmiennej a nie jej kopii (!!!)
 return liczba + kl.publiczna;
}

/// funkcja glowna programu
int main() {
 // funkcja statyczna jest dostępna nawet gdy nie ma rzadnego obiektu danej klasy
 int wynik = klasa2::funkcja1(4,8);
 printf("WYNIK: %d\n", wynik);

// deklaracje obiektów naszej klasy klasa1
 klasa1 k11;
 klasa1 k12 = * new klasa1(2);
 // k12 zostanie uwtorzona na stercie (a k11 na stosie)
 // w tym miejscu należy także wspomnieć o operatorach new i delete dla tablic,
 // które powinny być stosowane w C++ zamiast malloc'a:
 // char* tabliczka = new char[20]
 // w ten sposób zaalokowalibyśmy na stercie 20 bajtów
 // nalezy zaznaczyć że obiekty takie należy zwalniać przez
```

```
 // delete [] tabliczka;
 // operatory te mogą być stsowane też do bardziej złożonych obiektów
 // - wołają one odpowiednio konstruktory i destruktory tworzonych obiektów

// k11.prywatna=3; - tak nie mozna :-) bo prywatna

cout << "funkcja2: " << k11.funkcja2(31) << " " << k12.funkcja2(31) << endl;

k11.publiczna = 7;
 cout << "publiczna: " << k11.publiczna << " " << k12.publiczna << endl;

k11.wspolna = 3;
 cout << "wspolna: " << k11.wspolna << " " << k12.wspolna << endl;
 k12.wspolna++;
 cout << "wspolna: " << k11.wspolna << " " << k12.wspolna << endl;


/// wskaznik do klasy
 klasa1 * k1_wsk = &k11;
 cout << (*k1_wsk).funkcja1(11,56) << endl;

/// sprytne wskazniki
 auto_ptr<klasa1> k1_auto_wsk(new klasa1(22));
 cout << "funkcja2: " << k1_auto_wsk->funkcja2(100) << endl;
 // jak widać uzywa się ich jak normalne wskaźniki
 // różnica polega na tym iż utworzony obiekt typu klasa1 jest skojarzony
 // z tym wskaznikiem i zostanie automatycznie usuniety przy usunieciu wskaznika
 // dzieki czemu obiekt zostanie automatycznie usuniety gdy wyjdziemy z zasiegu w ktorym ⤸
 dostepny byl wskaznik
 //
 // wskazniki auto_ptr ze wzgledu na swoje zachowanie przy kopiowaniu
 // sa niebezpieczne gdy przekazujemy je do funkcji poprzez wartosc lub przechowujemy w ⤸
 kolekcji ...
 // rozwiazaniem tego problemu jest stosowanie boost:shared_ptr i boost:week_ptr
 // shared_ptr posiada licznik "kopiowań" dzieki czemu usuniety zostanie
 // dopiero gdy przestaniemy korzystac z ostatniej jego kopii

/// dziedziczenie
 klasa2 k21;
 cout << "funkcja1: " << k21.funkcja1() << endl;

klasa2 * k22 = new klasa2(10);
 cout << "funkcja2: " << k22->funkcja2(100) << endl;
 // tak tez mozna miec wskaznik do klasy
 // warto takze zauwazyc ze zamiast (*k22). mozemy uzywac k22->

cout << "jak widzimy funkcja2 zostala odziedziczona a funkcja 1 nadpisana ..." << endl;

// wywołanie obiektu jak funkcji - dzięki zdefiniowaniu operatora ()
 k21(3, 4);


/// wskaznik typu klasa1 moze wskazywac na obiekty pochodne od tej klasy
 k1_wsk = k22;
 // ale w takim wypadku nie dziala zwyczajne nadpisywanie funkcji
 cout << "funkcja1: " << k1_wsk->funkcja1() << " zachowuje sie jak z klasa1" << endl;
 // jednak dziala nadpisywanie gdy funkcje są deklarowane jako virtual
 cout << "funkcja3: " << k1_wsk->funkcja3() << " zachowuje sie jak z klasa2" << endl;
```

```
cout << "przeciazony operator: " << k21 + 12 << endl;
}

/* na koniec wspomnę o paru ciekawszych rozwiązaniach związanych z
     wykorzystaniem programowania obiektowego:
     * mechanizm klasy możemy użyć celem opakowania wywołania jakiejś
        funkcji wraz przekazywanymi do niej argumentami
        (w tym celu w klasie umieszczamy pole identyfikatora akcji,
        pola argumentów wywoływanej funkcji oraz metodę związaną z
        jej wykonaniem; dzięki temu że możemy tworzyć kolekcje obiektów
        takiej klasy możemy m.in. kolejkować zadania do wykonania,
        jak również zapamiętywać historię wykonanych poleceń - opcja undo),
        patrz wzorzec projektowy komendy
     * mechanizm dziedziczenia możemy wykorzystać do próby dynamizacji typów -
        operując klasa bazowa możemy w jakiś sposób wytworzyć (zależnie od
        potrzeb) jakiś obiekt którejś z klas pochodnych (o typie tworzonego
        obiektu mogą decydować czynniki nieznane w trakcie kompilacji - np.
        wartość odczytana z pliku), do wyboru mamy kilka sposobów
        implementacji tego typu mechanizmu - patrz wzorce projektowe fabryk
        obiektów oraz http://pl.wikipedia.org/wiki/RTTI
*/
```

SZABLONY.CPP

```
/** plik prezentuje podstawy programowania obiektowego w C++ :
 *    szablony
 *    przestrzenie nazw
 *    wyjatki
 **/


#include<iostream>
#include<limits>

using namespace std;

/// deklarujemy szablon klasy
// szablon może mieć kilka parametrów oraz parametry z wartością domyślną (opcjonalne)
template <typename T1, typename T2 = int> class klasa3 {
 T1 val; // gdy korzystamy z class domyslnie wszystko jest prywatne
 public:
  void set(T1 v);
  T1 get() {
   // gdyby parametr szablonu miałbyć jakąś klasą, która zawiera deklarację typu
   // i chielibyśmy odwołać się do tego typu, konieczne byłoby poinformowanie
   // kompilatora że odwołujemy się do typu, a nie składowej z tej klasy poprzez
   //      typename T3::typ_z_t3 var = 0;
   // (samo T3::typ_z_t3 szukałoby składowej typ_z_t3 w T3 ...)

   // niekiedy potrzebujemy sprawdzić czy przekazany do szablonu typ posiada
   // jakąś cechę (np. jak jest wartość maksymalna przekazanego typu liczbowego
   // w tym celu korzysta się z mechanizmu trait'ów - są to szablony prostych
   // funkcji zwracających pewną cechę przekazanego jako parametr szablonu typu
   // (np. czy jest to iterator, czy jest to tablica, ...)
   cout << "maksymalny zakres T2 wynosi " << numeric_limits<T2>::max() << endl;

   return val;
  }
```

```
  /// generowanie wyjątku
  void wyj(T2 i) throw (char const *, int) {
   switch(i) { // rzucamy wyjątek w zależnosci od i
    case 0:
     throw "Ala ma kota";
    case 1:
     cout << "wypisze sie (0) ?" << endl;
     throw "Kot ma Ale";
     // oczywiście gdy w definicji funkcji podaliśmy throw od innego obiektu
     // to tutaj po throw tez dajemy obiekt takiego typu
    case 2:
     throw i;
   }
   cout << "wypisze sie (1) ?" << endl;
  }
};

/// deklarujemy szablon funkcji z tej klasy
template <typename T1, typename T2> void klasa3<T1, T2>::set(T1 v) {
 val = v;
}

/// możemy zdeklarować także szablon funkcji nie należącej do klasy ...
// ponadto tutaj kompilator jest w stanie ustalić typ szablonu w oparciu
// o argument funkcji, dzieki czemu nie musimy jawnie podawać typu szablonu
// przy wywołaniu takiej funkcji ...
template <typename T1> T1 funkcja1(T1 x) {
 return 2 * x;
}

/// możemy tworzyć specjalizacje szablonów ...
template <> float funkcja1(float x) {
 return 3 * x;
}
// dla klasy wyglądałoby to tak:
//       template <> class klasa1<float> { ... }
// w przypadku pełnej specjalizacji - dla danego typu
// lub tak:
//       template <typename T1> class klasa1<T1*> { ... }
// w przypadku specjalizacji częściowej - dla dowolnego wskaźnika


/// z pomocą szablonów możemy także wykonywać obliczenia w trakcie kompilacji (np. silni)
template<int N> struct silnia {
 enum {val=N*silnia<N-1>::val};
};
template<> struct silnia <0>{
 enum {val=1};
};
///    parametrem szablonu nie musi być typ ...    może być wartość, a nawet inny szablon
//      nalezy tutaj zaznaczyć że jeżeli typ szablonu będącego parametrem
//      i typ naszego szablonu są powiązane (powinny być tego samego typu)
//      warto stosować następującą formę deklaracji, która pozwala uniknąć
//      błędu polegającego na podaniu sprzecznych typów
//        template<typename T, template<E, A> class C> class kont { ... }
//      wewnątrz klasy korzystamy z C<T, std::allocator<T>> jak z normalnego szablonu
//      a na zewnątrz wołamy kont<int, std:list> zamiast kont< int, std:list<int> >
//      gdy deklaracja byłaby: template<typename T, class C> class kont { ... }
//      takie przekazywanie szablonu może służyć do realizacji strategii w szablonach,
```

```
//      czyli decydowaniu na etapie wywołania jaką kację podejmuje szablon poprzez
//      podanie do niego odpowiedniego szablonu klasy realizującej taką akcję
//      (np. zapis do pliku, lub na stdout w systemie logowoania)

/// szablony mogą posłużyć także do korzystania z mechanizmu funkcji wirtualnych
// realizowanego na etapie kompilacji (bez dodatkowych narzutów w trakcie działania
// programu, jakie są związane z zwykłymi funkcjami wirtualnymi) ...
// realizuje się to poprzez dziedziczenie po szablonie klasy bazowej,
// w którym nasza klasa jest parametrm:
//      class abc : abecadlo_base<abc>
// a klasa bazowa implementuje funkcje wirtualne w formie wywołań:
//      return static_cast<T*>(this)->funkcja_wystepujaca_w_potomklach();
// podobny mechanizm jest często używany do uzupełniania klas potomnych o przeciążone
// operatory relacji realizowane przez negacje jednego operatora z klasy potomnej

/// szablony są niestety dość niewdzięczne w użyciu:
// - kod w nich podany jest kompilowany tylko wtedy gdy jest użyty
// - komunikaty błędów związane z użyciem szablonu na ogół wyglądają dziwnie i
//      wskazują gzieś w kod szablonu a nie użycie które wywołało błąd
//      problem ten można rozwiązać używając w szablonach asercji czasu kompilacji
//      np. BOOST_STATIC_ASERT(p) z boost/static_asert.hpp
// - rozrost kodu wynikowego, tutaj można próbować ograniczyć to zapewniając
//      współdzielenie kodu przez poszczególne szablony poprzez realizację za ich
//      pomocą adapterów dostosoójących typ do stosowanego przez klasę realizującą
//      obliczenia (np. zamieniających wartości na wskaźniki ...)



/// przestrzenie nazw
namespace przestrzen {
 void tescik() {
  cout << "przestrzen nazw" << endl;
 }
}

main() {
 /// tworzymy obiekty w oparciu o szablo9n klasy klasa3 dla typów double i int
 klasa3<double> d;
 klasa3<int> i;

d.set(13.497);
 i.set(14);

cout << d.get() + i.get() << endl;

cout << "funkcja1: " << funkcja1(12) << " " << funkcja1(12.5f) << endl;

/// przechwytujemy wyjątek
 for (int j=0; j<3; j++) {
  cout << endl << "j = " << j << endl;
  try {
   d.wyj(j);
   cout << "wypisze sie (2) ?" << endl;
   i.wyj(1);
  } catch (char const * opis) {
   cout << "WYJATEK: " << opis << endl;
  } catch (...) {
   cout << "inny wyjątek" << endl;
  }
  // aby zaoszczędzić jednego kopiowania należy przechwytywać wyjątek przez referencję
```

```
    // ale należy pamiętać iż ze względu na mechanizm działania wyjątków
    // wartość ta i tak będzie kopiowana
  }


/// korzystanie z przesztrzeni ...
  // przeszestrzenie stosowane standardowo (bez koniecznosci ich każdorazowego podawania)
  // deklaruje się przy pomocy dyrektywy "using namespace" (zobacz na początku pliku
  przestrzen::tescik();
}
```

NAPISY.CPP

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody operowania zmiennymi napisowymi
 *
 */

#include <stdio.h>

#include <sstream>
#include <string>
#include <iostream>

main() {
// najpierw jak to bylo w C ... czyli null-terminated string (NTS)
  // stala napisowa
  const char* text1 = "ala";
  printf("%s", text1);

// bufor napisowy o okreslonej długosci
  // - coś krótszego możemy przypisać przy incicjalizacji
  // możemy też pominąć 20 wtedy tablica ta będzie miała długość
  // odpowiadająca naszemu napisowi ...
  char text2 [20] = " ma";
  printf("%s", text2);

// badz np. przez sprintf()
  sprintf(text2, " kota\n");
  printf("%s", text2);

// rozszerzania GNU dodają równierz funkcje asprintf,
  // która dostaje wskaźnik, a pamięć potrzebą do stworzenia takiej
  // tablicy przydziela sama ...
  char *wsk = 0;
  asprintf(&wsk, "bla bla bla\n");
  printf("%s", wsk);

// natomiast nie pójdzie:
  // text2 = "test";

// można też z char* zrobić const char*
  const char* text3 = text2;
  printf("%s", text3);

// lista przydatnych funkcji operujących na takich napisach dostępna w: man 3 string
```

```
/* podsumowanie - char* (npis C)
   * wiele funkcji go lubi ... (zobacz np. man 3 string)
 */



// konwersja C++ <-> C
  // teraz deklaracja strumienia napisowego C++ i wpisanie do niej wartości
  std::ostringstream ostringstream_text;
  ostringstream_text << "cpp\n";

// zamiana na standardowy napis C - np. celem wypisania przez printf()
  printf("1) %s", ostringstream_text.str().c_str());

// można to zrobic rowniez tak ... (z wykorzystaniem zmiennej napisowej C++)
  std::string string_text1 = ostringstream_text.str();
  printf("2) %s", string_text1.c_str());
  // warto zaznaczyć że kompilator może optymalizować string przy przekazywaniu
  // do funkcji przez wartość jako "copy-on-write", co może spowodować problemy
  // przy programowaniu wielowątkowym
  // (gdy macierzysty zmodyfikuje zanim wątek zrobi własną kopię)

// a nawet tak ...
  int dlugosc = string_text1.length()+1;
  //    deklarujemy takblice - możnaby także użyć malloc
  char tresc1 [dlugosc];
  //    kopiujemy do niej nasz napis
  memcpy(tresc1, string_text1.c_str(), sizeof(char) * dlugosc);
  printf("3) %s", tresc1);

// albo i tak ...
  const char* tresc2 = NULL;
  tresc2 = ( ostringstream_text.str() ).c_str();
  printf("4) %s", tresc2);

// w powyższych punktach 1 do 4 przedstawiłem metody konwersji
  // ostringstream do napisu standardowego C,
  // zaczynając i kończąc w zasadzie na takiej samej metodzie ...

/* starszym sposobem byłoby tak (ale to juz przestazale ...):
 przed main():
   #include <strstream>
   using std::strstreambuf;
   using std::istrstream;
   using std::ostrstream;
   using std::strstream;
 i w main():
   ostrstream stream_text;
   stream_text << "test";
   stream_text.put(0);
   printf("%s", stream_text.str());
   stream_text.rdbuf()->freeze(0); // zwalniamy bufor
 */



// strumienie napisowe C++
  // możemy wydluzać tak utworzany napis:
  ostringstream_text << "abecadlo\n";
  std::cout << ostringstream_text.str(); //printf("%s", ostringstream_text.str().c_str());
```

```
// możemy także wykasować nasz napis:
 ostringstream_text.str(""); ostringstream_text.clear();
 std::cout << "nic tu _" << ostringstream_text.str() << "_ nie ma ?\n";

// teraz możemy znowu wpisywać
 ostringstream_text << "Wszyscy ludzie rodzą się wolni i równi pod względem swej ⤸
 godności";
 std::cout << ostringstream_text.str() << std::endl;

// można też połączyć te napisy
 std::ostringstream ostringstream_text2;
 string_text1 = "i swych praw.\n";
 ostringstream_text2 << ostringstream_text.str() << " " << 12.5 << string_text1;
 // i wpisać do string_text
 string_text1 = ostringstream_text2.str();
 std::cout << string_text1; //printf("%s", string_text2.c_str());

/* podsumowanie - ostringstream:
   * wygodnie się łączy i rozszerza
   * aby zamienic na string używamy metody .str()
   * aby utworzyć z czegokolwiek po prostu wpisujemy przez <<
     zatem wygodne do konwersji liczb na napis
     (patrz także boost::lexical_cast)

     więcej w in_out.cpp
 */



// zmienne napisowe C++
 // wyszukiwanie ...
 // pierwsze wystapienie zadanego ciągu
 int pozycja = string_text1.find(" sw");
 // możemy szukać od tyłu przez .rfind()
 if (pozycja != std::string::npos) {
  printf("znalazłem \" sw\" na pozycji %d\n", pozycja);
   // wyszukujemy od poprzedniej pozycji
  pozycja = string_text1.find(" sw", pozycja+1);
  if (pozycja != std::string::npos)
   printf("znalazłem kolejne \" sw\" na pozycji %d\n", pozycja);
 } else {
  printf("nie znalazłem \" sw\"\n");
 }

// pierwsze wystapienie zadanego któregoś ze znaku
 pozycja = string_text1.find_first_of(" i.");
 // możemy szukać od tyłu przez .find_last_of()
 // bądź też znaku innego niż podany odpowiednio przez
 //   .find_first_not_of() i .find_last_not_of()
 if (pozycja != std::string::npos)
  printf("znalazłem spację kropkę lub i na pozycji %d\n", pozycja);


// możemy wycinać zadany fragment napisu:
 std::string string_text2; // definicja drugiego napisu
 string_text2 = string_text1.substr(73); // od 74 znaku (włącznie) do końca
 string_text1 = string_text1.substr(0,14); // 14 znaków od początku
```

```
// możemy też łączyć napisy typu string bez pośrednictwa ostringstream
 string_text1 += " " + string_text2;
 // można też przy pomocy metod zamiast przeciążonych operatorów
 //    string_text.append(" ").append(string_text2);
 // co da identyczny rezultat
 std::cout << string_text1;

std::cout << "Wpisz coś długiego\n";
 std::getline( std::cin, string_text2 );
  // wczyta całą linię z standardowego wejścia do string_text
  // jako trzeci argument możemy podać inny ogranicznik niż znak nowej linii
 std::cout << "Wpisałeś: " << string_text2 << std::endl;

string_text1 = "Ala ma kota";
 string_text2 = "Ala ma psa";
 // możliwe jest także porównywanie fragmentów napisów:
 // string_text1.compare(od_1, ilosc_1, string_text2, od_2, ilosc_2)
 if (string_text1.compare(1, 5, string_text2, 2, 5))
  printf("podnapis 2-7 (numerujemy od zera) jest taki sam\n");
 if (string_text1 != string_text2)
  printf("ale napisy są różne\n");

// wstawianie - insert(pozycja, co)
 string_text2.insert(6, " kota i");
 std::cout << string_text2 << std::endl;

// zastępowanie - replace(pozycja, ile, czym);
 string_text2.replace(4, 2, "miała samochód", 0, 6);
 // mogłoby tez być string_text2.replace(4, 2, "miała"); i parę innych wariantów ...
 std::cout << string_text2 << std::endl;

// usówanie - erase(pozycja, ile);
 string_text2.erase(9, 1); // 9 zamiast 8 bo UTF-8 i ł ma dwa znaki
 std::cout << string_text2 << std::endl;

// zmiana długości - resize(długość, znak_dopełniający)
 string_text2.resize(14);
 std::cout << string_text2 << std::endl;

string_text2.resize(18, 'x');
 std::cout << string_text2 << std::endl;

/* podsumowanie - string
   * wygodny do operowania - wyszukiwanie, wycinanie, ...
   * łączenie i rozszerzenie też całkiem dobre ;-)
   * aby zamienic na char* (npis C) używamy metody .c_str()
 */

// oraz na koniec - podzial linii na pola:
 std::string linijka = "12 14 25 ";
 int tmp=-1, old_tmp, x;
 for (int i=0; i<3; i++) {
  // ustalamy granice pola
  old_tmp = tmp + 1;
  tmp = linijka.find(" ", old_tmp);
  // konwertujemy pole na liczbe
  x = atoi(linijka.substr(old_tmp, tmp).c_str());
  // wypisujemy
  std::cout << "Pole numer: " << i << " ma wartosc: " << x << std::endl;
 }
```

```
}
```

STL.CPP

```
/*
 * w napisy.cpp przedstawiłem już podstawy korzystania z <string> i <sstream>
 * w tym programie przedstawię kilka kolejnych elementów biblioteki standardowej C++
 *
 * zobacz także:
 *    http://pl.wikipedia.org/wiki/Standard_Template_Library
 *    http://pl.wikibooks.org/wiki/C++#Dodatek_A:_Biblioteka_STL
 *    http://www.intercon.pl/~sektor/cbx/std/stl.html
 */

#include <vector>
#include <list>
#include <map>
#include <algorithm>

#include <string>
#include <boost/bind.hpp>
#include <boost/lambda/lambda.hpp>

#include <iostream>
#include <stdio.h>

using namespace std;

void wypisz(pair<string, string> p) {
 printf("YY %s : %s\n", p.first.c_str(), p.second.c_str());
}

void wypisz2(int a, int b) {
 printf("QQ %d : %d\n", a, b);
}

int sum(int a, int b) {
 return a+b;
}

struct por {
 int a;
 int b;
 bool operator<(const por &p) { return a < p.a; };
 bool operator>(const por &p) { return a > p.a; };
};

main() {
 int i=0;

// uwaga: wszystkie kontenery STL operują na kopiach obiektów do nich wstawianych
 // wszystkie konternery mogą zawierać dowolny typ - tak wbudowany jak i definiowany
 // przez użytkownika, dla działania funkcji sortujących konieczne może być zapewnienie
 // operatora większości dla takiego typu

/// dynamiczna tablica czyli "vector"
 // uwaga: powiększanie vectora może doprowadzić do konieczności jego przeniesienia w ⤸
 pamięci
```

```
  // operacja ta związana jest z kopiowaniem wszystkich elementów wektora w nowe miejsce
  // operacja taka jest kosztowana czasowo
  // dlatego lepiej aby vector przechowywł wskaźniki na duże elementy a nie duże elementy
  // warto także przed rozpoczęciem wypełniania vectora określić rozmiar
  // jaki chcemmy na niego zarezerwować przy pomocy metody reserve

// wskaźnik na element wektora może stracić aktualność na sklutek usunięcia elementów
  // wcześniejszych niż wskazywany lub wspomnianej wyżej realokacji wektora
  // podobnie rzecz ma się z innymi kontenerami bazującymi na tablicach

// tablica 7 liczb całkowitych wypełniona jedynkami
 vector<int> tablica(7,1);

// iterator (służy do wskazywania miejsca w tablicy)
 vector<int>::iterator iter;

// wstawiamy przed 3 (pierwszy +2) elementem wartość 33
 iter = tablica.begin();
 iter += 2;
 tablica.insert(iter, 1, 33);
 // uwaga powoduje to przekopiowanie wszystkich elementów od iter włacznie

// zmieniamy wartość 6 elementu na 7
 tablica[5]=7;
 // zwiększamy wartość 8 (w chwili obecnej ostatniej ostatniego, bo 7 początkowych + 1)
 // i wypisujemy ten element
 printf("XX: %d\n", ++tablica[7]);

// wstawiamy na końcu
 tablica.push_back(15);

// usuwamy element 7
 iter = tablica.begin();
 tablica.erase(iter + 6);
 // uwaga powoduje to przekopiowanie wszystkich elementów od iter włacznie

// wypisanie całej tablicy
 for (iter; iter != tablica.end(); ++iter) {
  printf("%d : %d\n",i++,*iter);
 }

// wskaźnik na początkowy i końcowy element tablicy
 // uwaga wartości wskaźników do elementów mogą być nietrwałe - przekopiowywanie ↩
elementów
 int *ref_f = &(tablica.front());
 int *ref_b = &(tablica.back());
 printf("XX: %x %x %d\n", ref_f, ref_b, ref_b-ref_f+1);

// gdy nie pusty
 if ( ! tablica.empty() )
  // wypisz aktualny, maksymalny bez przenoszenia w inne miejsce,
  // maksymalny wogole rozmiar
  printf("%ld %ld %ld\n", tablica.size(), tablica.capacity(), tablica.max_size());

// usuwamy ostatni element
 tablica.pop_back();

// usuwamy wszystkie elementy
 tablica.clear();
```

```
// oczywiście są także inne przydatne metody tego obiektu ...

/// kolejka o dwóch końcach - deque
  // jest to kontener umożliwiający wydaje wstawianie elementów na koniec i na początek
  // implementowany jest jako zbiór tablic, dzięki czemu dodanie elemntu nigdy nie powoduje
  // przepisywania całości, pozatym bardzo przypomina vector



/// listy czyli "list"
  list<int> lista;

// wstawiamy coś na listę (można wstawiać także na wybraną pozycję - jak w vector)
  lista.push_back(30);
  lista.push_back(20);
  lista.push_back(29);
  lista.push_back(40);
  lista.push_back(-50);
  lista.push_back(20);
  // można też z przodu (a także gdzieś w środku - insert jak w vector)
  lista.push_front(29);

// usuwamy z listy elementy równe 29
  lista.remove(29);

// sortujemy naszą listę (można także określić funkcję zadającą porządek
 sortowania)
  lista.sort();

// usuwamy występujące koło siebie duplikaty
  lista.unique();

// odwracamy listę
  lista.reverse();

// wypisujemy naszą listę
  // do elementów listy możemy odwoływać się tylko przez iteratory (nie da się przez
 indeks)
  for(list<int>::iterator iter=lista.begin(); iter != lista.end(); ++iter)
    printf("%d ",*iter);
  puts(" << normalna");

for(list<int>::reverse_iterator iter=lista.rbegin(); iter != lista.rend(); ++iter)
    printf("%d ",*iter);
  puts(" << odwrocona");
  // zastosowanie rbegin i rend jest wygodniejsze niz próba uzycia w tym celu petli
  // od end do begin gdyz begin wskazuje na pierwszy element, a end na pierwszy
  // za ostatnim (sa one niesymetryczne co utrodnia takie proste odwrocenie)


// oczywiście są także inne przydatne metody tego obiektu ...
  // możemy jeszcze m.in. usuwać ciągi elementów bądź przenosić je w ramach listy ...

/// listy jednokierunkowe "slist"
  // odmienny wariant list (możemy tylko zwiększać iteratory)
  // zajmują mniej pamięci i są szybsze za wyjątkiem   insert, erese i splice
  // (bo działają na elemencie przed podanym,
  //     dlatego dostępne dla tych list są szybsze odpowiedniki - metody *_after)
```

```
/// tablice haszujące (asocjacyjna) - STL map
 map<string,string> tab_asoc;
 tab_asoc["Ala"]="ma kota";
 tab_asoc["kot"]="ma Ale";

printf("Ala %s\n", tab_asoc["Ala"].c_str());



/// inne kontenery
  // w ramach STL dostępne są też inne obiekty - mniej lub bardziej podobne do ⤸
 przedstawionych powyżej
  // np. zapewniające uporządkowanie ich elementów (set, map, ..)


/// algorytmy
  // dostępne są także częstostosowane wzorce funkcji operujące na kontenerach zwane ⤸
 ”algorytmami STL”
  // nalezy jednak pamiętać że jeżeli możemy skorzystać z metody kontenera lub z ⤸
 algorytmu
  // należy wybrać to pierwsze, gdyż ta implementacja ma duże szanse być sporo ⤸
 wydajniejsza

printf("%d %d\n", min(10, 30), min(30, 10));

// inną przydatną funkcją jest for_each() powodujący wykonywanie podanej
  // funkcji na elementach pomiędzy dwoma itrerałami
 for_each(tab_asoc.begin(), tab_asoc.end(), wypisz);

// warto wspomnieć też o samym std:pair użytym w funkcji wypisz:
 wypisz( pair<string,string>("AA", "BB") );
  // jest to szablon łączący dwa obiekty zadanych typów w parę ... (taki jednoelementowy ⤸
 map)

// przy korzystaniu z algorymów często zachodzi potrzeba dostosowania wywołania ⤸
 istniejącej funkcji
  // lub metdoy do wymaganego przez algorytm, zadanie to ułatwiają dostarczone przez ⤸
 bibliotekę
  // standardową adaptery takie jak bind2st, mem_fun, ... jednak wygodniejszym od nich ⤸
 sposobem
  // jest użycie boost::bind
 boost::bind(wypisz2, _2, _1)(1, 3);
 boost::bind(wypisz2, _1, _1)(1, 3);
 (boost::bind(wypisz2, 10, _1))(1, 3);
  // wewnątrz możemy podstawiać kolejne funkcje przy pomocy bind’a ...
  // ponadto zawsze możemy wskazać jawnie typy atrgumentów funkcji o jaką nam chodzi ...
  // poprzez podanie w tym wypadku <int,int>
 boost::bind(wypisz2, _3, boost::bind<int,int>(sum, _1, _2))(1, 3, 8);

// a teraz praktyczny przyklad z sortwaniem wektora
 vector<por> pors(2);
 pors[0].a=4; pors[0].b=10;
 pors[1].a=13; pors[1].b=2;

//sort(pors.begin(), pors.end());
 printf("posortowane po a: %d %d\n", pors[0].a, pors[1].a);

sort(pors.begin(), pors.end(), boost::bind(&por::b,_1) < boost::bind(&por::b,_2) );
  // w przypadku wołania metod lub składowych za _1 zostaje podstawiony obiekt którego ⤸
```

```
metodę wołamy
 printf("posortowane po b: %d %d\n", pors[0].a, pors[1].a);

// nieco podobnym do boost::bind rozwiązaniem jest boost::lambda
  // umożliwia on tworzenie nienazwanych obiektów funkcyjnych
  // poprzez ich definiowanie "w miejscu uzycia"
  cout << "Lista: ";
  for_each(lista.begin(), lista.end(), cout << boost::lambda::_1 << ' ' );
  cout << endl;

// z innych ciekawych tego typu rzeczy warto wspomniec o boost::function implementującym ⤸
 wzorzec
  // komendy (przechowywanie wskaźnika na funkcję w formie obiektu) dzięki niemu możliwe ⤸
 jest
  // przechowywanie w kolekcjach STL ciągów wywołań funkcji (np. dla funkcji "undo")


/// boost rozszerza możliwości STL także w zakresie kontenrów - implementuje m.in.:
  // * tuple - "rozszerzone pair" umozliwiające przechowywanie zestawu wielu elementów, a ⤸
 nie tylko pary
  // * variant - "unia dla obiektowów" (klasyczna unia z C nie może przechowywać obiektów ⤸
 klas)
  // * optional - oprócz wartości może mieć też stan "niezainicjalizowany"
  // * tribool - logika trójwartościowa
  // * any - taki odpowiednik dla void*
  // * array - adapter dla tablic z C dostarczający metod typowych dla kolekcji STL
  // * multi_array - tablice wielowymiarowe
  // jest także biblioteka boost do obsługi grafów - Boost Graph Library


/// wracając do algorytmów nalezy jeszcze wspomnieć przynajmniej o:
  // - find (zwraca iterator do pierwszego elementu spełniającego warunek)
  // - count (zwraca liczbę elementów spełniających warunek
  // - remove (usuwa wskazane elementy (a tak na prawde przenosi pozostale elementy na ⤸
 poczatek,
  //              elementy usuwane moga byc zamazane ale miejsce po nich pozostaje
  //              - zmienia sie tylko wartośc końca ...)
  // - partition (dzieli zbiór na spełniające i niespełniające warunek
  // wszystkie funkcje sortujące bez static_ na początku mogą zmieniać kolejnośc ⤸
 elementów
  // o tym samym porządku sortowania - spełniających ten sam warunek (np. elementów ⤸
 równych)

// w ramach STL zaimplementowane są różnego typu algorytmy
  // odsyłam do: http://www.sgi.com/tech/stl/table_of_contents.html
  // lub do: http://pl.wikibooks.org/wiki/C++/Algorytmy_w_STL
}
```

Podstawy programowania Unixa/Linuxa

- katalogi.c - katalogi, czyli nasze własne ls ;-) + obsługa błędów z wykorzystaniem errno.h (mechanizm bardzo przydatny przy korzystaniu z bardziej zaawansowanych funkcji ... w celu ustalenia dlaczego to nie działa)
- unix.c - funkcje Unix'owate ... (w tym sygnały, deskryptory i czas)
- opcje.c - przetwarzanie opcji unix'owych (-h) i GNU (–help)
- fork.c - wywołania innych programów i pipe
- watki.c - tworzenie aplikacji wielowątkowych z wykorzystaniem pthread
- watki_clone.c - tworzenie aplikacji wielowątkowych z wykorzystaniem clone()

— semafory_posix.c - używanie semaforów, timerów, pamięci współdzielonej oraz asynchronicznego wyjścia zgodnych ze standardem POSIX[994]owych
— semafory_ipc.c - wykorzystywanie semaforów i innych mechanizmów SystemV IPC[995] (svipc) do komunikacji między procesami,
więcej o komunikacji międzyprocesorowej w LINUX - Synchronizacja i komunikacja procesów[996], Distributed Systems[997]
— tty-listener.c - program wykorzystujący urządzenia vcs i vcsa do podebrania ekranu (tekstu) z terminala tty

KATALOGI.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje metodę odczytu katalogów
 *
 */


// będziemy coś wypisywać ...
#include <stdio.h>

// funkcja exit
#include <stdlib.h>

// katalogi
#include <dirent.h>

// obsługa błędów
#include <errno.h>

main() {
 DIR* katalog = opendir( "/tmp" ); // otwieramy katalog
 // obsługa błędów - np nie istniejącego katalogu
 if (katalog == NULL) {
  // nie udało się - zapamiętujemy kod błędu
  int kod_bledu=errno;

  // mamy parę metod wypisania ...
  perror( "próba otwarcia katalogu" );
  fprintf(stderr, "błąd (kod %d): \"%s\" (%s : %d)\n", kod_bledu, strerror(kod_bledu), ⤸
  __FILE__, __LINE__);

  // kończymy zwracając informację o niepowodzeniu (1) do rodzica
  exit (1);

  // warto takze zapoznac sie z funkcjami z rodziny: err(), warn(), a także error() i syslog(⤸
  )
  // w szczególności uzyteczny jest w nich output formatowany printf z obsługą %m (⤸
  wwstawia komunikat z ernno)
  //
  // error(-1, 0, "Ala ma %d kotow: %m", 5); lub errx("Ala ma %d kotow: %m", 5);
  //        wypisze komunikat podstawiając za %m aktualny błąd errno i zakonczy progam z kodem ⤸
  -1
  // error(0, 0, "Ala ma %d kotow: %m", 5); lub warnx("ooo %m ppp %i", 5);
  //        wypisze komunikat podstawiając za %m aktualny błąd errno ale nie zakończy ⤸
  programu
  // error(0, 1, "Ala ma %d kotow: %m", 5);
  //      wypisze komunikat podstawiając za %m błąd odpowiadjący errno=1
 }
```

```
printf("typ\tnazwa\n");
 struct dirent* pozycja = 0; // tworzymy strukturę odpowiedzialną za przechowywanie ⤸
 informacji o pozycji katalogu
 while (pozycja = readdir(katalog)) { // pętla po kolejnych pozycjach katalogu
  printf("%d\t%s\n", (*pozycja).d_type, (*pozycja).d_name);
 }

closedir(katalog); // zamknięcie katalogu

// warto także spomnieć o POSIXowych funkcjach do operacji na plikach i katalogach takich ⤸
 jak:
  //        rename, chmod, chown, mkdir
  // należy zaznaczyć tutaj iż nie ma funkcji systemowej kopiującej plik
  //        - kopiowanie odbywa się przez otwarcie odczyt w pętli i zapis pod nową nazwą

exit (1);
}
```

UNIX.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje użyteczne funkcje charkterystyczne dla systemów zgodnych    z Unix'em
 *
 */

#include <stdio.h>
#include <signal.h>
#include <fcntl.h>
#include <time.h>

// funkcje obsługi sygnału
void sig_usr(int sig) {
 printf("Otrzymałem sygnał numer: %i\n", sig);
}

void akcja_sygnalu(int sig, siginfo_t *info, void *context) {
 printf("Otrzymałem sygnał numer: %i z kodem %d i wartością %d\n",
  sig, info->si_code, info->si_value.sival_int);
 // si_code okresla powód wysłania sygnału
}

main() {
 printf("Mój PID to %d\n", getpid());

// obsługa sygnałów
  // do zdefiniowania obsługi sygnału - jego przechwycenia służy funkcja
  //signal( numer_sygnału, sposób_obsługi );
  // użytkownik ma do dyspozycji dwa nie zdefiniowane przez system sygnały: SIGUSR1, SIGUSR2
  // może też zmieniać obsługę innych (poza nieprzechwytywalnymi) ...

// aby wysłać sygnał do samego siebie korzystamy z raise( sygnał );
  // co jest równoważne kill( getpid(), sygnał );
  // parę słów o sygnałach zawarłem też w fork.c

// ustawiam prechwycenie sygnalu SIGUSR1 (pełną ich listę wypisuje kill -l)
  // lista z opisami dostepna na http://en.wikipedia.org/wiki/Signal_(computing)⤸
 #List_of_signals
```

```c
 signal(SIGUSR1, &sig_usr);
 signal(SIGUSR2, &sig_usr);
 // zamiast wskaźnika do funkcji możemy podac także SIG_IGN albo SIG_DFL
 // co oznacza odpowiednio ignorowanie tego sygnału lub jego domyślną obsługę


// możliwe jest też maskowanie sygnałów
 sigset_t maska_sygnalow;
 if (sigemptyset(&maska_sygnalow)) // inicjalizujemy na pustą, na pełną byłoby sigfillset(
)
  puts("sigemptyset error");
 if (sigaddset(&maska_sygnalow, SIGINT)) // dodajemy sygnał do maski
  puts("sigaddset error");
 // można także usuwać - sigdelset() i sprawdzac czy jest obecny - sigismember()
 if (sigprocmask(SIG_SETMASK, &maska_sygnalow, NULL))
  puts("sigprocmask error");
 // w skutek tego program nie będzie reagował na Ctrl+C
 //
 // oprocz tego sa funckcje API sygnałowgo BSD:
 //    sigvec, sigblock, sigsetmask, siggetmask, sigmask

// dostępny jest także machanizm kolejkowanej obsługi sygnałów
 // w którym także kolejne sygnały tego samego typu w oczekiwaniu
 // na odebranie gromadzone są w kolejce
 struct sigaction akcja_obslugi_sygnalu;
 akcja_obslugi_sygnalu.sa_flags = SA_SIGINFO;
 sigfillset(&akcja_obslugi_sygnalu.sa_mask);
 akcja_obslugi_sygnalu.sa_sigaction = &akcja_sygnalu;
 sigaction(SIGALRM, &akcja_obslugi_sygnalu, NULL);

sigqueue(getpid(), SIGALRM, (const union sigval)123);


// możliwe jest także oczekiwanie na dowolny sygnał:
 puts("czekam na sygnal");
 pause();

// lub na sygnał z zestawu okreslonego w taki sposób jak dla
 // poprze sigwait(&zestaw, &sygnal);
 sigemptyset(&maska_sygnalow);
 sigaddset(&maska_sygnalow, SIGQUIT);
 int signum;
 puts("czekam na sygnal SIGQUIT");
 sigwait(&maska_sygnalow, &signum);
 if (signum == SIGQUIT)
  puts("dostalem SIGQUIT");

// możemy sobie zarzyczyć także przesłania sygnału SIGALRM
 // za określoną ilość sekund (lub troche później):
 alarm(4);

// funkcja open służy do otwierania dostępu do pliku przez deskryptor, patrz: man 2 open
 int deskryptor = open( "/tmp/plik", O_WRONLY );

write(deskryptor, "Ala ma Kota", 12 ); // wpisuje 12 znaków z podanego ciągu
 // do czegoś okreslonego deskryptorem (w tym przykładzie jest to plik)

// mamy też bliźniaczą funkcję read(deskryptor, bufor, ilosc) czytająca z deskryptora
 // do bufora zadana ilosc znaków oraz funkcję lseek(deskryptor, przesuniecie, tryb)
 // - przesówjącą się w pliku; tryb to SEEK_SET (nowa pozycja = przesuniecie),
```

```
  // SEEK_CUR (nowa pozycja = aktualna pozycja + przesuniecie),
  // SEEK_END (nowa pozycja = rozmiar_pliku + przesuniecie)

close(deskryptor); // zamykamy deskryptor

// mamy równierz trochę innych funkcji:
  // chown - zmiana użytkownika i grupy
  // chdir - zmaina aktualnego katalogu (odpowiednik komendy cd ...)
  // getcwd - zwraca aktualny katalog, tak jak getenv("PWD");
  // rmdir - usuwa pusty katalog

// czas ...
 time_t czas = time(0);
   // czas mierzony w sekundach od początku epoki (czyli 1970-01-01 00:00:00 UTC),
   // patrz: man 2 time
   // możemy też korzystać z wywołania: time(&czas);
 printf("Od poczotku epoki upłyneło %d sekund\n", czas);

// następnie możemy utworzyć struktórę zawierającą czas rozłożony:
 struct tm czas_rozlozony_utc;
 gmtime_r(&czas, &czas_rozlozony_utc);
 // jest też localtime zwracająca czas w lokalnej strefie czasowej oraz inne funkcje
 //    patrz man 3 gmtime
 struct tm czas_rozlozony_local;
 localtime_r(&czas, &czas_rozlozony_local);
 // uwaga na wywołanie czas_rozlozony_local=localtime(&czas);
 // może nadpisać ono strukturę czas_rozlozony_utc
 // (zwróci wskaźnik do tego samego obszaru co poprzednie wywołanie)

char czas_napis [100];
 // następnie możemy czas ten sformatować przy pomocy strftime
 strftime(czas_napis, sizeof(czas_napis), "%Y-%m-%d %H:%M:%S %Z (%s)", &czas_rozlozony_utc);
   // więcej o napisach formatujących - man 3 strftime
 printf("Mamy teraz (UTC): %s\n", czas_napis);

strftime(czas_napis, sizeof(czas_napis), "%Y-%m-%d %H:%M:%S %Z (%s)", &czas_rozlozony_local);
 printf("Mamy teraz (local): %s\n", czas_napis);

// albo odwołać się bezpośrednio do składowych struktury tm
 printf("Aktualna godzina w UTC to: %d\n", czas_rozlozony_utc.tm_hour);

// wato zwrócić uwagę iż niektóre z pozycji napisu formatującego funkcji strftime ⤸
 korzystają z
  // strefy czasowej ustalonej w zmiennej środowiskowej co ma niemiłe skutki w połączeniu ⤸
 z gmtime()
  // jak widać na powyższym przykładzie timestamp dla czas_rozlozony_utc jest nie poprawny ⤸
 gdyż
  // funkcja ta zakłada że podana struktura jest czasu lokalnego u nas != UTC

// aby zobaczyć to lepiej możemy zrobić to dla chwili zero
  // (która powinna odpowiadać 1970-01-01 00:00:00 UTC
 czas = 0;
 gmtime_r(&czas, &czas_rozlozony_utc);
 localtime_r(&czas, &czas_rozlozony_local);
  // należy pamiętać że funkcję powyzsze w wariancie bez _r są niebezpieczne przy ⤸
 wielowątkowości

strftime(czas_napis, sizeof(czas_napis), "%Y-%m-%d %H:%M:%S %Z (%s)", &czas_rozlozony_utc);
 printf("Mamy teraz (UTC): %s\n", czas_napis);
 strftime(czas_napis, sizeof(czas_napis), "%Y-%m-%d %H:%M:%S %Z (%s)", &czas_rozlozony_local);
```

```
 printf("Mamy teraz (local): %s\n", czas_napis);

// widzimy już że problem jest w funkcji wykonywanej na strókturze z rozłożonym czasem ⤸
 w UTC
 setenv("TZ", "UTC");
 strftime(czas_napis, sizeof(czas_napis), "%Y-%m-%d %H:%M:%S %Z (%s)", &czas_rozlozony_utc);
 printf("Mamy teraz (UTC): %s\n", czas_napis);
 strftime(czas_napis, sizeof(czas_napis), "%Y-%m-%d %H:%M:%S %Z (%s)", &czas_rozlozony_local);
 printf("Mamy teraz (local): %s\n", czas_napis);
 // daje to porpawny rezultat dla UTC oraz niepoprawny dla lokalnego ...
 // co ciekawe strefa czasowa (%Z) zawsze jest poprawna, należy więc uważać
 // gdy korzystamy z gmtime oraz napisów formatujących zależynych od strefy czasowej ...

// mamy równierz funkcję mktime,
 //    która na podstawie struktury tm wyrazonej w czasie lokalnym tworzy timestamp

// oraz funkcję gettimeofday pobierającą czas z większą dokładnością
 struct timeval czas2;
 gettimeofday(&czas2, NULL);
 printf("Od poczotku epoki upłyneło %d sekund i %d microsekund\n", czas2.tv_sec, czas2.⤸
 tv_usec);
 // podobnie do gettimeofday działa clock_gettime(CLOCK_REALTIME, &time);


while(sleep(60) > 0); // aby móc pobawić się sygnałami ...
   // while konieczny bo sygnal przerywa czekanie i nie jest ono kontynuowane
   // należy też wspomnieć iż ładniejszą metodą realizacji takiego zabronienia
   // zakończenia programu jest czekanie na jakis sygnal niz robienie sleep()
   // ale tu nam chodzi aby ten sleep sie kiedys zakonczyl ...

   // sleep(ilosc_sekund); - proces robi sobie przerwę przez zadaną liczbę sekund lub ⤸
   dłużej
   //    (nie zajmuje czasu procesora w odróżnieniu od rozwiązań typu pętle itp.,
   //    otrzymanie sygnału skraca przerwę)
}
```

OPCJE.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje korzystanie z funkcji parsujących ospcje programu
 *
 * zobacz tez: man getopt_long
 *
 */

// dla wypisywania:
#include <stdio.h>

// dla obsługi opcji:
#include <getopt.h>

// dla CHAR_MAX:
#include <limits.h>

// deklaracje numerów wyboru dla opcji długich nie posiadających krótkich odpowiedników
// CHAR_MAX + 1 => numery zaczynają się tam gdzie kończą się numery
// odpowiadające kodom pojedynczych znaków ...
```

```
enum {
 OPCJA_PIERWSZA = CHAR_MAX + 1,
 OPCJA_DRUGA,
 OPCJA_TRZECIA
};

// struktura przechowujaca informacje o opcjach programu ...
// nazwa_dluga, czy_wymaga_argumentow, zmienna_do_ustawienia, nazwa krotka
static struct option opcje [] =
{
 {"wymaga-argumentu", required_argument, NULL, 'w'},
 {"moze-miec-argument", optional_argument, NULL, OPCJA_PIERWSZA},
 {"kolejna", no_argument, NULL, OPCJA_DRUGA},
 {"i-jeszcze-jedna", no_argument, NULL, OPCJA_TRZECIA},
 {0, no_argument, NULL, 'v'},
 {0, 0, 0, 0}
};

int main(int argc, char *argv[]) {
 int opcja;
 while((opcja = getopt_long(argc, argv, "w:vh", opcje, 0)) != -1) {
  switch (opcja) {
   case 'v':
    printf("Podana opcja -v\n");
   break;
   case 'w':
    printf("Podana opcja -w z argumentem %s\n", optarg);
   break;
   case OPCJA_PIERWSZA:
    printf("Podana opcja --moze-miec-argument z argumentem %s\n", optarg);
    if (!optarg)
     printf("  bez argumentu\n");
   break;
   case OPCJA_DRUGA:
    printf("Podana opcja --kolejna\n");
   break;
   case OPCJA_TRZECIA:
    printf("Podana opcja --i-jeszcze-jedna\n");
   break;
   default:
    printf("nie podano opcji ...\n");
  }
  printf("  zczytałem opcje do argv[%d] wlacznie\n", optind );
 }

printf("XXX: %d\n", wyswietl_pomoc);

return 0;
}
```

FORK.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje rozgałęzianie programu i komunikację międzyprocesową
 *
 */
```

```
#define _GNU_SOURCE
// (lub _XOPEN_SOURCE) dla WEXITSTATUS

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>

void zakonczenie() {
 fprintf(stderr ,"Postanowiłem skończyć ... PID %d\n", getpid());
}

main(int argc, char *argv[], char *envp[]) {

// w zasadzie nalezaloby okreslic funkcje obslugujaca sygnal informujacy o smierci potomka
 // i wywolujaca wtedy funkcje z waitpid(-1, NULL, WNOHANG); ... inaczej beda powstawaly
pocesy zombie
 // jak to zrobic pokazane jest w dsp_gui_gtk.c

pid_t pid, parent_pid;
 parent_pid = getpid();

// ustalamy jaka funkcja ma być wykonywana przed exit(); nie będzie wykonywana przed abort(
);
 // która brutalniej kończy program
 atexit( zakonczenie );

// rozgałęziamy proces
 switch (pid = fork()) {
  case -1: // funkcja zwróciła -1 co oznacza błąd
   fprintf(stderr, "Error in fork\n");
   return 1;

  case 0: // funkcja zwróciła zero co oznacza że jesteśmy w procesie potomnym
  {
   printf("Jestem procesem potomnym. Moj PID = %d, PID mojego rodzica to: %d\n", getpid(),
   parent_pid);

   // mogę wykonać jakiś program i dalej kontynuować
   int retcode = system ("ls -l");
   printf("Nadal żyję !!! A komenda odpalona w system() zwróciła kod powrotu: %d\n",
   WEXITSTATUS(retcode));

   // do uruchamioania zewnętrznego programu można także spróbować użyć posix_spawnp()
   // działa ona na zasadzie połączenia fork() i exec() ... więcej info w man

   char* katalog_domowy = getenv("HOME");
    // pobieram zawartość zmiennej środowiskowej HOME (określającej katalog domowy)
   printf("moim katalogiem domowym jest: %s\n", katalog_domowy);

   // dostęp do zmiennych środowiskowych możliwy jest także przez tablicę będącą
   // trzecim argumentem funkcji main:
   while (*envp) { // pętla wypisuje wszystkie przekazane do programu zmienne środowiskowe
    printf("%s\n", *envp++);
  }

   // można również tworzyć zmienne środowiskowe ...
   //   setenv("nazwa zmiennej", "tresc", nadpisz);
   // nadpisz==0 nie nadpisuję ewentualnej istniejącej takiej zmiennej
```

```
    // nadpisz==1 nadpisuje
    setenv("HOME", "/tmp", 1);
    // uwaga: zmianie ulega tylko środowisko aktualnego programu i programów przez niego
    //              wywoływanych a nie systemu (inne procesy nie odczują zmiany)
    // oraz usuwać:
    //    unsetenv("fork_testowa"); // usuwam utworzoną przez siebie zmienną środowiskową

    printf("a teraz moim katalogiem domowym jest:\n");
    // mogę też zastąpić siebie jakimś programem
    execlp("printenv", "printenv", "HOME", 0);
    printf("a teraz zostałem zastąpiony innym programem i się nie wypiszę :-(\n");

    exit(0);
   }

   default:  // funkcja zwróciła coś innego od 0 i -1 oznacza to że jestesmy w pocesie ⤸
   macierzystym
     // i otrzymaliśmy pid naszego dziecka ...
   {
    printf("Jestem procesem macierzystym. Moj PID = %d, PID mojego potomka to: %d\n", getpid(),⤸
    pid);

    // poczekam aż potomek się zakończy ...
    // warto zauważyć że dopóki nie będę czekał wykonuje się współbiernie z ⤸
    potomkiem ...
    // w szczególności nawet mógłbym się zakończyć ...
    // rodzicem potomka powinien stac się wtedy init (proces o PID==1)
    wait(0);
    printf("potomek już zrobił swoje ...\n");
   }
  }


// tworzymy łącze nienazwane - rurę ...
 int deskryptory_rury1 [2];
 if ( pipe(deskryptory_rury1) == -1 ) fprintf(stderr, "Error in pipe\n");
 //deskryptory_rury1[0] - odbiór z rury
 //deskryptory_rury1[1] - wysyłanie przez rurę

// znowu rozgałęziamy proces
 switch (pid = fork()) {
 case -1:  // funkcja zwróciła -1 co oznacza błąd
  fprintf(stderr, "Error in fork\n");
  return 1;

  case 0:
  {
   dup2(deskryptory_rury1[1], 1);  // podmieniamy deskryptor 1 (stdout) na koniec rury do ⤸
   pisania
   // dla stdin, sdtou i stderr mamy domyślnie otwarte deskryptory o numerach ustalonych na ⤸
   sałe
   // nazwa strumienia in/out - numer deskryptora
   // stdin - 0
   // stdout - 1
   // stderr - 2
   close(deskryptory_rury1[1]);  // zamykmy bo skopiowaliśmy na stdout
   close(deskryptory_rury1[0]);  // zamykamy bo piszemy a nie czytamy
   system ("uname -a");

   pause();  // zatrzymuje program w oczekiwaniu na sygnał
```

```
    while (1); // to i tak sie nie wykona bo sygnałem będzie SIGTERM

    return 0;
   }

   default:
   {
    FILE *rura = fdopen(deskryptory_rury1[0], "r"); // tym razem kojarzymy z rurą obiekt typu ⤸
    FILE ...
    // identyczny efekt możnaby uzyskać przez
    //FILE *rura =   popen( "uname -a", "r" );
    // jednak ten w odróznieniu od prezentowanego tutaj nie umożliwia skonstruowania
    //   komunikacji dwukierunkowej

    char napis [20];
    fgets( napis, 20, rura ); // pobieram z rury 20 znaków ...
    printf("%s\n", napis); // i wypisuję

    kill(pid, SIGTERM); // prosimy potomka by się zakończył ... poprzez wysłanie sygnału (⤸
    więcej w unix.c)
    return 1;
   }
  }
}
```

WATKI.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje tworzenie wątków i uruchamianie w nich funkcji ...
 *
 * więcej o wątkach: http://rainbow.mimuw.edu.pl/SO/Linux/Temat02/opis_watki.html
 *
 * kompilacja: gcc -lpthread -o watki watki.c
 *
 */

#define _GNU_SOURCE
// to potrzebne aby flagi zachowywały się   zgodnie z POSIX

#include <pthread.h>
#include <sched.h>
#include <unistd.h> /* for sleep */
#include <stdio.h> /* fo printf */
#include <stdlib.h> /* for malloc */
#include <bits/errno.h> /* for ENOTSUP */

int globalna; // zmienna ta wykorzystana jest do zademonstowania możliwosci korzystania
// ze zmiennych globalnych do komunikacji między wątkami
// uwaga: zmienne globalne są alokowane w obszarze pamięci procesu a nie na stosie
//         jak ma to miejsce ze zmiennymi lokalnymi

int wypisz(char * napis) {
 char i;
 globalna = 1;
 for(i=0; i<4; i++) {
  printf("To watek: %s\n", napis);
```

```
  sleep(1);
}
 globalna = 0;
 sleep(5);
 pthread_exit(0); // koncze dzialanie watku
}

void wypisz2(char * napis) {
 printf("ten wątek umiera ... %s\n", napis);
}

int wypisz3(char * napis) {
 int i;
 printf("To watek: %s\n", napis);

// zabraniamy przerwania watku w trakcie ponizszego wypisywania
 pthread_setcancelstate(PTHREAD_CANCEL_DISABLE, NULL);
 for(i=0; i<10; i++) {
  printf(".", napis);
  fflush(stdout);
  sleep(1);
}
 printf("\n", napis);
 // a teraz juz moze byc anulowywany
 pthread_setcancelstate(PTHREAD_CANCEL_ENABLE, NULL);

// ale od teraz tylko w wyznaczonych momentach
 // na funkcjach takich jak sleep, system, ...
 // czy tez pthread_testcancel()
 pthread_setcanceltype(PTHREAD_CANCEL_DEFERRED, NULL);

sleep(50);
}

pthread_mutex_t mut;
pthread_cond_t war;
int licz;

int wypisz4(char * napis) {
 int i;
 printf("To watek: %s\n", napis);

sleep(2);

// blokujemy wejście na ten kawałek kodu
 if (pthread_mutex_trylock(&mut)) {
  printf("nie mogłem zrobić lock więc czekam do skutku\n");
  pthread_mutex_lock(&mut);
}

printf("jestem w sekcji krytycznej - przygotowuje dane\n");
 sleep(5);
 licz++;

// zezwalam na odbiór
 if (licz>0) pthread_cond_signal(&war);
 pthread_mutex_unlock(&mut);
}
```

```
main () {
 int i;

// tworzymy nowy watek uruchamiajac w nim funkcje wypisz
 // i przekazując jako pierwszy argument tej funkcji ”Ala ma kota”
 // - to generuje ostrzezenie ale dziala ...
 globalna = -1;

pthread_t watek1;
 pthread_attr_t atrybuty;
 pthread_create( &watek1, 0, (void* (*)(void *))wypisz, "Ala ma kota" );

// pobranie atrybutu wątka i sprawdzenie czy jest wątkiem procesu czy systemu
 pthread_getattr_np(watek1, &atrybuty);
 pthread_attr_getscope(&atrybuty, &i);
 printf("PTHREAD_SCOPE: %d %d %d\n", i, PTHREAD_SCOPE_SYSTEM, PTHREAD_SCOPE_PROCESS);
 // jezeli typ == PTHREAD_SCOPE_SYSTEM to watek jest systemowy czyli bedzie widoczny w ps -m
 system("ps -m");

for(i=0; i<7; i++) {
  printf("To program główny (%d)\n", globalna);
  sleep(1);
}

// możemy oczekiwać na zakończenie wątku poprzez wywołanie w innym funkcji pthread_join ⤸
 ,
 // umożliwa ona także uzyskanie kodu z jakim zakończył się wątek
 printf("Program główny czeka na koniec wątku\n");
 pthread_join(watek1, NULL);
 printf("Wątek zakończył się\n");
 // jej wywołanie jest konieczne aby wątek po zakończeniu był w całości usunięty
 // chyba ze zostal okreslony jako taki do ktorego nie mozna sie dolaczyc
 // przez wywolanie pthread_detach() lub podanie przy tworzeniu flagi PTHREAD_CREATE_DETACHED

// przygotowujemy atrybuty dla wątku
 if (pthread_attr_init(&atrybuty))
  perror("pthread_attr_init");
 // ustawiamy że ma być to wątek procesu
 i = pthread_attr_setscope(&atrybuty, PTHREAD_SCOPE_PROCESS);
 if (i != 0) {
  fprintf(stderr, "pthread_attr_setscope ERROR %d\n", i);
  if (i == 95) // ENOTSUP
   fprintf(stderr, "  wątki procesu nie sa wspierane\n");
}
 // możemy ustawiać takie rzeczy jak to czy wątek jest typu odłączonego
 // (czy będziemy mogi czekać na niego przy pomocy join czy nie) - ⤸
 pthread_attr_setdetachstate
 // parametry schedulera, adres stosu watku, itd - więcej info w man 3 pthread_attr_init

// okreslamy funkcje wykonywana w momecie smierci watka
 pthread_cleanup_push((void (*)(void *))wypisz2, "byłem wypisz3");

  // tworzymy inny watek
 pthread_create(&watek1, &atrybuty, (void* (*)(void *))wypisz3, "do anulowania");

pthread_getattr_np(watek1, &atrybuty);
 pthread_attr_getscope(&atrybuty, &i);
 printf("PTHREAD_SCOPE: %d %d %d\n", i, PTHREAD_SCOPE_SYSTEM, PTHREAD_SCOPE_PROCESS);
 system("ps -m");
```

```
sleep(5);
 puts("postanowilem zabic ten watek");

   // i go konczymy z zewnatrz
   // w zależności od ustawień wątku anulowanie wątku dzieje się natychmiastowo
   // albo w jednym z punktów anulowania - zobacz funkcje to ustawiające w wypisz3()
 pthread_cancel(watek1);
   // wypada też odebrać jego kod powrotu
 pthread_join(watek1, NULL);
 puts("a teraz watek juz niezyje, ale może jeszcze po sobie sprzątać ...");

// odefiniowujemy funkcje wykonywana w momecie smierci watka
 pthread_cleanup_pop(1);

// biblioteka obsługująca wątki zapewnia także mechanizm zapewniający pojedyńcze ⤸
 wykonanie
  // jakiejś czynności (tylko jeden z wątków (pierwszy) przedwarzających wspólny kod
  // wykona ta czynność - zobacz man pthread_once
  //
  // a także implementuje funkcje umożliwiające wątkom korzystanie z semaforów oraz ⤸
 sygnałów - zobacz
  // man pthread_mutex_init , man pthread_sigmask , man pthread_cond_init
  // możemy korzystać także z semaforów biblioteki standardowej C - zobacz man 3 sem_init

pthread_mutexattr_t atrybuty_mutexu;
 pthread_mutexattr_settype(&atrybuty_mutexu, PTHREAD_MUTEX_ERRORCHECK);
  // możemy ustawić PTHREAD_MUTEX_ERRORCHECK aby mechanizm reagował błędem na ⤸
 unlockowanie
  // zalokowanego przez inny wątek mutexu itp, PTHREAD_MUTEX_RECURSIVE aby blokada nalozona
  // przez watek A nie obowiazywala watku A (umożliwia to korzystanie z mutexów w funkcjach
  // rekurencyjnych), albo PTHREAD_MUTEX_NORMAL lub PTHREAD_MUTEX_DEFAULT ...
  // flagi te mogą się nieco różnić w zaleznośći od stosowanej biblioteki i użytych ⤸
 #define
  // szczegółów należy szukać w plikach features.h pthread.h oraz dokumentacji man

pthread_mutexattr_setprotocol(&atrybuty_mutexu, PTHREAD_PRIO_NONE);
  // poza nic nie zmieniającym PTHREAD_PRIO_NONE mamy także PTHREAD_PRIO_INHERIT ⤸
 zabezpieczający
  // wątki przed inwersją priorytetów (priorytet po wejściu do sekcji krytycznej jest ⤸
 podwyższany
  // do najwyższego z priorytetów procesów czekających oraz PTHREAD_PRIO_PROTECT ⤸
 zabezpieczający
  // również przed zakleszczeniem dzięki podnoszeniu tego priorytetu powyżej najwyższego ⤸
 z
  // priorytetów procesów czekających
  // do zapobiegania inwersji priorytetów przydatne jest też: ⤸
 pthread_mutexattr_setprioceiling();

// inicjalizujemy mutex
  // dla domyślnych ustawień można podać NULL jako drugi argument
 pthread_mutex_init(&mut, &atrybuty_mutexu);

pthread_create(&watek1, &atrybuty, (void* (*)(void *))wypisz4, "z mutexem");

// blokujemy mutex
 pthread_mutex_lock(&mut);

printf("jestem w sekcji krytycznej - czekam na dane\n");

// czekamy ...
```

```
 while (licz == 0) pthread_cond_wait(&war, &mut);
  // pthread_cond_wait zwalnia tak na prawdę na chwile podany do niego mutex a po odebraniu
  // pthread_cond_signal na dozorowanym przez niego warunku ponownie go zamyka

printf("jestem w sekcji krytycznej - odbieram dane\n");
 sleep(5);

licz--;
 // odblokowujemy mutex
 pthread_mutex_unlock(&mut);

// niszczymy mutex
 pthread_mutex_destroy(&mut);


// wiecej w man 7 pthreads
 return 0;
}
```

WATKI_CLONE.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje tworzenie wątków (poprzez clone) i uruchamianie w nich funkcji ...
 *
 * więcej o wątkach: http://rainbow.mimuw.edu.pl/SO/Linux/Temat02/opis_watki.html
 *
 * kompilacja: gcc -o watki watki_clone.c
 *
 */

#include <pthread.h>
#include <sched.h>
#include <unistd.h> /* for sleep */
#include <stdio.h> /* fo printf */
#include <stdlib.h> /* for malloc */

int globalna; // zmienna ta wykorzystana jest do zademonstowania mozliwosci korzystania
// ze zmiennych globalnych do komunikacji między wątkami

int wypisz(char * napis) {
 char i;
 globalna = 1;
 for(i=0; i<4; i++) {
  printf("To watek: %s\n", napis);
  sleep(1);
 }
 globalna = 0;
 return 0; // koncze dzialanie watku
}

main () {
 char i;

globalna = -1;

// jest to metoda rozgaleziania procesu (zblizona do funkcji fork())
 // charakterystyczna dla Linuxa, w odroznieniu od pthread oba watki beda
```

```
  // widzoczne jako osobne procesy, mozna to zmienic podajac flagi
  // CLONE_SIGHAND | CLONE_THREAD,
  // podana tuatj flaga CLONE_VM oznacza wspoldzielenie pamieci,
  // przy pomocy tych flag mozna jeszcze wiecej ...
  // wiecej o flagach mozna przeczytac w man 2 clone

// jest takze funkcja sys_clone ktora kontynuuje program
  // od miejsca wywolania a nie od zadanej funkcji

void *stos;
 stos = (void *) malloc(100);
 // na wiekszosci architektur na ktorych dziala Linux stos rosnie w dol,
 // wiec podajemy koniec przydzielonego zakresu pamieci
 clone(wypisz, stos+99, CLONE_VM, "Ala juz nie ma kota");

for(i=0; i<7; i++) {
   printf("To znowu program główny (%d)\n", globalna);
   sleep(1);
 }
 return 0;
}
```

SEMAFORY_POSIX.C

```
/**
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje korzystanie z pamięci współdzielonej SHM oraz semaforów POSIXowych
 *
 * kompilacja gcc -lrt semafory_posix.c
**/

#include <sys/stat.h>

#include <fcntl.h>
#include <sys/mman.h>

#include <semaphore.h>
#include <time.h>

#include <mqueue.h>

#include <signal.h>

#include <aio.h>

#include <errno.h>
#include <stdio.h>

void sig_usr(int sig) {}

int main(int argc, char *argv[]) {
 /// pamięć dzielona przez shm i mmap (wiecej man 7 shm_overview)
 // mozna także przez mmap i zwykły plik - patrz alokacja_pamieci.c
 int fd=shm_open("moja_pamiec", O_RDWR|O_CREAT, 0770);
 int size=ftruncate(fd,sizeof(int), SEEK_SET);
 void *addr=mmap(0,sizeof(int), PROT_READ|PROT_WRITE, MAP_SHARED, fd, 0);

int *licznik = (int*) addr;
```

```
  // jeżeli więcej jakiś danych to najlepiej w tablicy lub w strukturze
  // no chyba że ktoś lubi liczyć ręcznie offsety ...

printf("Licznik (1): %d\n", (*licznik)++);
 printf("Licznik (2): %d\n", (*licznik)++);
 // tak na prawdę to należałoby operacje na tym liczniku pozabezpieczać semaforami ...

sleep(5);

munmap(addr,sizeof(int));
 close(fd);
 shm_unlink("moja_pamiec");


/// semafory POSIX (wiecej man 7 sem_overview)
 sem_t *semafor;

// otwieramy semafor nazwany
 // zamiast tego mozemy po prostu uzyc struktury typy sem_t
 // w takim wytpadku przed pierszym uzyciem musimy wykonac na niej sem_init
 // a po zakonczeniu uzywania przez ostatni proces sem_unlink
 semafor = sem_open("/moj_semafor", O_CREAT, 0770, 1);
 // flaga O_CREAT powoduje ze semafor zostanie utworzony gdy nie istnieje
 // oraz wymusza podanie dwoch dodatkowych argumentow funkcji ...
 // dodanie O_EXCL w flagach spowodowaloby ze funkcja zworci blad
 // gdy semafor o takiej nazwie istnieje
 //
 // eleganckim wydaje się używanie
 // sem_open("moj_semaforek", O_CREAT|O_EXCL, ...); w procesie glownym
 //    wraz z obsluga tworzenia innej nazwy gdy dostaniemy EEXIST
 // a w "klientach" sem_open("moj_semaforek", 0);

if (semafor == SEM_FAILED) {
  perror("sem_open");
  return -1;
 }


if (sem_trywait(semafor)) {
  puts("Nie udało się opóścić semaforu");
  perror("sem_trywait");

  struct timespec timeout;
  clock_gettime(CLOCK_REALTIME, &timeout);
  // funkcja o dzialaniu zblizonym do gettimeofday ..

  // zwiekszamy aktualny czas o 5s
  timeout.tv_sec+=5;

  if (sem_timedwait(semafor, &timeout)) {
   perror("sem_timedwait");
   // zamykamy semafor - nie bedziemy z niego korzystac
   sem_close(semafor);
   goto kolejki;
  }
 }
 // jest tez oczywiscie sem_wait która czeka do skutku ...

// śpimy, to nieładnie spać w sekcji krytycznej, ale to tylko demo ...
 puts("jesteśmy w sekcji krytycznej");
```

```
 sleep(10);

// podnosimy semafor
 sem_post(semafor);

puts("wyszliśmy z sekcji krytycznej");

// usuwamy semafor nazwany, ma to wplyw na wszystkie procesy z niego korzystajace
 sem_unlink("/moj_semafor");



/// kolejki wiadomosci POSIX (wiecej man 7 mq_overview)
 mqd_t kolejka;

struct mq_attr atrybuty;
kolejki:
 atrybuty.mq_maxmsg=3; // pojemność kolejki - 3 wiadomości
 atrybuty.mq_msgsize=20; // po 20 bajtów każda

// otwarcie kolejki
  // dla kolejki o domyślnych parametrach można podać NULL jako ostatni argument
  // w drugim argumencie można podać
  //    O_NONBLOCK można podać aby kolejka była nie blokująca
  //    O_EXCL - działanie takie samo jak w sem_open()
  // trzeba podać tryb (read, write, read+write)
 kolejka = mq_open("/moja_kolejka", O_CREAT | O_RDWR, 0777, &atrybuty);
 if (kolejka == (mqd_t) -1) {
  perror("mq_open");
  return -1;
 }

char buf[100]; // mogly byc 20 bo tyle wynosi mq_msgsize, ale potem sie nam przyda ...
 unsigned priorytet;
 struct timespec timeout;
 clock_gettime(CLOCK_REALTIME, &timeout);
 timeout.tv_sec+=1;

// odbieramy wiadomość z timeoutem ...
 size = mq_timedreceive(kolejka, buf, 100, &priorytet, &timeout);
 // mamy także możliwość odbioru bez timeoutu bądź zarządania powiadomienia
 // o wpisaniu pierwszej wiadomości do niepustej kolejki - mq_notify()
 if (size > 0)
  printf("odebralem %d bajtow z priotytetem %d: %s\n", size, priorytet, buf);
 else
  perror("mq_timedreceive");

// wysyłamy wiadomość do kolejki
 // funkcja ta zawiesi proces gdy nie ma miejsca w kolejce która nie jest
 // otwarta z O_NONBLOCK, istnieje też wariant z timeoutem
 mq_send(kolejka, "Ala ma kota", 12, 9);
 // kolejka moze miec zarowno wielu producentow jak i odbiorcow
 // raz odebrana wiadomosc znika z kolejki
 //
 // wiadomosci odbierane beda w kolejnosci priorytetow
 // a w ramach jednego priorytetu w kolejnosci wysylania
 //
 // w przypadku braku miejsca w kolejce procesy wysyłające są zawieszane

sleep(5);
```

```
mq_unlink("/moja_kolejka");



/// timery POSIX (wiecej man timer_settime)
 // przygotowujemy strukture opisujaca zdarzenie zwiazane z tyknięciem timera
 // taki sam mechanizm zdarzen wykorzystywany jest w spomnianej wcześniej mq_notify()
 struct sigevent zdarzenie_timera;
 zdarzenie_timera.sigev_notify = SIGEV_SIGNAL; // zostaniemy poinformowani sygnałem
 // może być także SIGEV_THREAD wtedy w sigev_notify_function podajemy funkcje
 // ktora ma utworzyc nowy watek, w sigev_notify_attributes atrybuty wątku a w
 // sigev_value.sival_ptr argument funkcji wątku ... przyklad w man 3 mq_notify
 zdarzenie_timera.sigev_signo = SIGPWR; // będzie to sygnał SIGPWR
 zdarzenie_timera.sigev_value.sival_int = 13; // wartość sygnału

// musimy zglosic ze jestesmy zainteresowania obsluga SIGPWR
 signal(SIGPWR, &sig_usr);

// tworzymy timer
 timer_t licznik_czasu;
 timer_create(CLOCK_REALTIME, &zdarzenie_timera, &licznik_czasu);
 // gdyby zamiast zdarzenie_timera podać NULL to jestesmy budzeni sygnałem SIGALRM

// konfiguryujemy i uruchamiamy timer
 struct itimerspec zaleznosci_czasowe;
 // pierwsze wykonanie timera po 100ms
 zaleznosci_czasowe.it_value.tv_sec=0;
 zaleznosci_czasowe.it_value.tv_nsec=100000000;
 // kolejne co 1s
 zaleznosci_czasowe.it_interval.tv_sec=1;
 zaleznosci_czasowe.it_interval.tv_nsec=0;
 timer_settime(licznik_czasu, 0, &zaleznosci_czasowe, NULL);
 // warto zwrocic uwage na flage TIMER_ABSTIME umozliwiajaca ustawienie timera
 // na czas absolutny (a nie okres czasu)
 // jezeli it_interval bedzie wyzerowane to timer wykona się tylko raz
 char i=0;
 do {
  struct itimerspec pozostalo_do_timera;
  timer_gettime(licznik_czasu, &pozostalo_do_timera);
  printf("do timera posostalo: %dns\n", pozostalo_do_timera.it_value.tv_nsec);
  pause(); // czekamy na timer
  printf("aktualna liczba utracionych tykniec timera wynosi %d\n",
   timer_getoverrun(licznik_czasu));
 } while(i++<4);
 // powyzsza metoda gwarantuje rozpoczynanie w rownych odstepach czasu ...
 // w przypadku checi zastosowania sleep() należałoby obliczac czas ktory sie
 // wykonywalismy i odejmowac go od czasu ktory podajemy do sleep()

// robimy sobie drzemmke ...
 struct timespec drzemka, pobudka;
 drzemka.tv_sec=2;
 drzemka.tv_nsec=100000000;
 // chemy spac 2.1s ale obudzi nas timer
 // to ile zesmy niedospali zostanie zapisane w pobudka
 if (nanosleep(&drzemka, &pobudka)<0 && errno == EINTR)
  printf("niedospalismy: %ds i %dns\n", pobudka.tv_sec, pobudka.tv_nsec);

// usuniecie timera
 timer_delete(licznik_czasu);
```

```c
/// asynchroniczne IO
 struct aiocb opis_aio;
 opis_aio.aio_fildes = open( "/etc/passwd", O_RDONLY); // czytamy plik
 opis_aio.aio_offset = 0; // od poczatku
 opis_aio.aio_buf = (void *) buf; // do bufora buf
 opis_aio.aio_nbytes = 100; // ile czytamy
 opis_aio.aio_sigevent.sigev_notify = SIGEV_NONE;
 // nie korzystamy z powiadomienia o koncu
 // jest to taki sam mechanizm jak dla timerow ...
 opis_aio.aio_reqprio = 1; // priorytet operacji

if (aio_read(&opis_aio))
  perror("aio_read");

// czekamy na koniec operacji io
 // w ogolnosci mozemy podac cala liste operacji io
 // - funkcja wychodzi gdy ktorakolwiek sie z nich zakonczy
 struct aiocb *lista_aio[1];
 lista_aio[0] = &opis_aio;
 aio_suspend(&lista_aio, 1, NULL);

// odbieramy stan zakonczenia
 size = aio_error(&opis_aio);
 if (size == 0) {
  size = aio_return(&opis_aio);

  buf[99]='\0';
  printf("Wczytałem %d bajtów: %s\n", size, buf);
 } else {
  printf("aio_error return %d\n", size);
 }
}
```

SEMAFORY_IPC.C

```c
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje korzystanie z komunikacji międzyprocesowej poprzez SystemV IPC (svipc):
 * semafory pamięć współdzielona i kolejki komunikatów
 *
 * warto zajrzeć też na: http://rainbow.mimuw.edu.pl/~mengel/jpp/sem/
 *
 */

#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/shm.h>

#include <stdio.h>
#include <errno.h>
#include <string.h>

// z manula dotyczacego semctl:
 #if defined(__GNU_LIBRARY__) && !defined(_SEM_SEMUN_UNDEFINED)
```

```
  /* union semun jest zdefiniowana w <sys/sem.h> */
#else
  /* dla zgodnoci z X/OPEN musimy sami sobie zdefiniować */
union semun {
 int val;                        /* wartość dla SETVAL */
struct semid_ds *buf;           /* bufor dla IPC_STAT i IPC_SET */
unsigned short *array;          /* tablica dla GETALL i SETALL */
                                 /* Część specyficzna dla Linuksa: */
struct seminfo *__buf;          /* bufor dla IPC_INFO */
};
 #endif


int main(int argc, char *argv[]) {
 int pid=getpid();

/// nazwa i klucz semafora
 const char * nazwa_semafora = "Moje_ICP";
 key_t klucz=0;
 int i;
 for (i=0; nazwa_semafora[i]!=0; i++)
  klucz += nazwa_semafora[i];
 // w zasadzie to do tworzenia klucza należałoby użyć funkcji ftok(char *pathname, char ⤸
 proj_id);

printf("%d: uzyskałem klucz: %x\n", pid, klucz);


/** SEMAFORY **/


/// tworzymy zestaw semaforów bądź uzyskujemy dostęp do istniejącego
 // tworzymy semafor gdy nie istnieje
 int identyfikator_zestawu_semaforow = semget (klucz, 1, 0640|IPC_CREAT|IPC_EXCL);
  // pierwszym argumentem jest dluga liczba calkowita pelniaca funkcje nazwy
  // zestawu semaforow (zamiast niej mozna podac IPC_PRIVATE wtedy jednak
  // zestawu możemy używać tylko w tym i w potomnych procesach)
  // dugim jest liczba semaforow w zestawie
  // trzecim są flagi (okreslajace prawa dostepu do zestawu semaforow
  // (jak do pliku) oraz sposob otwarcia (IPC_CREAT - utworzenie gdy nie ma
  // lub otwarcie gdy jest, IPC_CREAT | IPC_EXCL - utworzenie gdy nie ma,
  // błąd gdy jest)
 if (identyfikator_zestawu_semaforow != -1) {
 // jeżeli się udało nadajemy mu wartość początkową
  printf("%d: utworzyłem zestaw semaforów o identyfikatorze: %i\n", pid, ⤸
  identyfikator_zestawu_semaforow);
  union semun ctl_arg;
  ctl_arg.val=1;
  semctl (identyfikator_zestawu_semaforow, 0, SETVAL, ctl_arg);
   // drugim argumentem jest numer semafora
   // trzecim argumentaem jest polecenie
   // (SETVAL = ustaw na wartość z ctl_arg.val, więcej man 2 semctl)
   // trzecim jest union semun w tym wypadku interesuje nas pole val
 } else if (errno == EEXIST) {
 // w przeciwnym razie korzystamy z istniejącego
  identyfikator_zestawu_semaforow = semget (klucz, 1, 0640);
  printf("%d: uzyskałem dostęp do semafora o identyfikatorze: %i\n", pid, ⤸
  identyfikator_zestawu_semaforow);
 } else {
  printf("BŁĄD: %i", errno);
```

```
  return -1;
 }

/// uzywanie semaforów
 // - sprawdzamy czy jest zezwolenie (ewentualnie czekamy na nie) i ustawiamy blokadę

// tablica struktur opisujących operacje
 struct sembuf operacje [1];
 // pierwsza operacja
 (operacje[0]).sem_num = 0; // pierwszy, czyli zerowy ;-) semafor
 (operacje[0]).sem_op = -1; // gdy wartość semafora >= |-1| dodaj do wartości
                            // semafora -1 i kontynuuj, inaczej czekaj ...
 (operacje[0]).sem_flg = SEM_UNDO; // możliwe są:
 //    SEM_UNDO (zmiana stanu semafora cofnieta w momęcie zakończenia procesu) i
 //    IPC_NOWAIT (gdy nie może być wykonana natychmiast konczy semop z błędem)

printf("%d: oczekiwanie na semafor\n", pid);

// wykonanie operacji
 semop (identyfikator_zestawu_semaforow, operacje, 1);
  // trzecim argumentem jest liczba operacji do wykonania na zestawie
  // semaforów (ilość elementów tablicy z argumentu drugiego)
  // operacje te wykonane będą w sposób niepodzielny ...
  // zobacz też: man 2 semop

printf("%d: jestem za semaforem\n", pid);
 printf("%d: naciśnij enter aby zwolnic semafor\n", pid);
 while (getchar() != 10);

(operacje[0]).sem_op = 1; // zwiększamy wartość semafora (zwalniamy)
 (operacje[0]).sem_flg = 0; // nie chcemy cofnięcia tej operacji
 semop (identyfikator_zestawu_semaforow, operacje, 1);

printf("%d: semafor zwolniony\n", pid);

/// usuwanie semafora
 (operacje[0]).sem_op = -2; // zwiększamy wartość semafora (zwalniamy)
 (operacje[0]).sem_flg = IPC_NOWAIT; // nie chcemy cofnięcia tej operacji
 if (semop (identyfikator_zestawu_semaforow, operacje, 1) != -1) {
  semctl (identyfikator_zestawu_semaforow, 0, IPC_RMID, 0);
  printf("%d: usuwam semafor\n", pid);
 } else {
  printf("%d: nie usuwam semafora\n", pid);
 }


/** PAMIEC WSPOLDZIELONA **/
// tworzymy lub podłączamy się do fragmentu pamięci współdzielonej
 int deskryptor_pamieci_dzielonej = shmget(klucz, 100, 0640|IPC_CREAT | SHM_R | SHM_W);
 if (deskryptor_pamieci_dzielonej < 0) {
  printf("BŁĄD shmget: %i (%s)\n", errno, strerror(errno));
  return -1;
 }
 void * adres_pamieci_wspolnej = (void *) shmat(deskryptor_pamieci_dzielonej, NULL, 0);
 if ((int) adres_pamieci_wspolnej == -1) {
  printf("BŁĄD shmat: %i (%s)\n", errno, strerror(errno) );
  return -1;
 }

// czytamy, zwiększamy o 1 i zapisujemy
```

```
int *wartosc, old, new;
wartosc = adres_pamieci_wspolnej;
old = *wartosc;
new = old + 1;
*wartosc = new;
printf("odczytałem %i, ustawiłem %i\n", old, new);

// gdy wartosc > 5 usuwamy
if (new > 5) {
 printf("Zlecam usunięcie wspólnego segmentu pamięci\n");
 // odłączamy od naszej przestrzeni adresowej
 shmdt(adres_pamieci_wspolnej);
 // zaznaczamy do usunięcia
 shmctl(deskryptor_pamieci_dzielonej, IPC_RMID, NULL);
}


/** KOLEJKI KOMUNIKATOW **/


// struktura opisująca komunikaty
struct {
 long typ;
 char tresc[20];
} komunikat;

// uzyskujemy identyfikator kolejki komunikatów
int identyfikator_kolejki_komunikatow = msgget(klucz, 0640|IPC_CREAT);
if ( identyfikator_kolejki_komunikatow < 0) {
 printf("BŁĄD msgget: %i (%s)\n", errno, strerror(errno) );
 return -1;
}


// odbieramy komunikat ... jeżeli nie ma to nie czekamy
if ( msgrcv(identyfikator_kolejki_komunikatow, &komunikat, sizeof(komunikat), 2, IPC_NOWAIT) ⤸
< 0 ) {
 if (errno != ENOMSG) {
  printf("BŁĄD msgrcv: %i (%s)\n", errno, strerror(errno) );
  return -1;
 } else {
  printf("Brak komunikatów\n");

  // wysylamy komunikat
  komunikat.typ=2;
  strcpy(komunikat.tresc, "Hello");
  if ( msgsnd(identyfikator_kolejki_komunikatow, &komunikat, sizeof(komunikat), 0) < 0) {
   printf("BŁĄD msgsnd: %i (%s)\n", errno, strerror(errno) );
   return -1;
  }
  printf("Nadano komunikat: %s\n", komunikat.tresc);
 }
} else {
 printf("Otrzymano komunikat: %s\n", komunikat.tresc);

 printf("Usuwam kolejkę komunikatów\n");
 // usuwamy kolejke komunikatów
 msgctl(identyfikator_kolejki_komunikatow, IPC_RMID, NULL);
}
```

```
return 0;
}
```

TTY-LISTENER.C

```
/*
 Prosta implementacja programu podsłuchującego wybrany terminal tty.

 Działająca wg poniższego schematu:
 width=`od -j1 -N1 -d /dev/vcsa1 | cut -b8-`
 fold -w $width /dev/vcs1 > /tmp/new
 if ! diff /tmp/new /tmp/old; then
   mv /tmp/new /tmp/old
   render /tmp/old
 fi
*/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <inttypes.h>
#include <sys/ioctl.h>
#include <linux/vt.h>

#define DEVICE_SURFIX 2

int getConsole(char devSurfix, uint8_t *w, uint8_t *h) {
 int fd;
 char tmpBuf[20];

snprintf(tmpBuf, 20, "/dev/vcsa%d", devSurfix);
 printf("Console on %s ", tmpBuf);

fd = open(tmpBuf, O_RDONLY);
 if (fd < 0) {
  perror("open: ");
  exit(1);
 }
 if (read(fd, tmpBuf, 4) != 4) {
  perror("read: ");
  exit(2);
 }
 close(fd);

*h = tmpBuf[0];
 *w = tmpBuf[1];
 printf(" has %d rows and %d cols\n", *h, *w);


snprintf(tmpBuf, 20, "/dev/vcs%d", devSurfix);
 printf("Openning console on %s \n", tmpBuf);

fd = open(tmpBuf, O_RDONLY);
 if (fd < 0) {
  perror("open: ");
  exit(3);
 }
```

```
return fd;
}

int main() {
 int vcsFD, myFD;
 struct vt_stat vtstat;

uint8_t width, height, y;
 ssize_t sizeIN, sizeOUT;
 char    *bufIN_1, *bufIN_2, *bufOUT, *bufTMP, charTMP;

vcsFD = getConsole(DEVICE_SURFIX, &width, &height);
 myFD = open("/dev/tty0", O_RDONLY);

sizeIN  = width * height;
 bufIN_1 = malloc(sizeIN);
 bufIN_2 = malloc(sizeIN);
 sizeOUT = (width+1) * height;
 bufOUT  = malloc(sizeOUT);

*bufIN_2 = '\0';

while (1) {
  ioctl(myFD, VT_GETSTATE, &vtstat);
  if (vtstat.v_active == DEVICE_SURFIX) { // run only when DEVICE_SURFIX is active VT
   lseek(vcsFD, 0, SEEK_SET);
   if (read(vcsFD, bufIN_1, sizeIN) == sizeIN) {
    if ( strncmp(bufIN_1, bufIN_2, sizeIN) != 0 ) { // redraw only when changed VT output
     bufTMP = bufIN_1;
     for (y=0; y<height; y++) {
      // save first char of next line and put in its place "end of string"
      charTMP = bufTMP[width];
      bufTMP[width] = '\0';

      // show line
      puts(bufTMP);

      // set bufTMP to start of next line and restore first char of next line
      bufTMP = bufTMP + width;
      *bufTMP = charTMP;
     }

     // swap bufIN_2 and bufIN_1
     bufTMP  = bufIN_2;
     bufIN_2 = bufIN_1;
     bufIN_1 = bufTMP;
    }
   }
  }
  usleep(100000);
 }
}
```

Sieci TCP/IP

— numeracja_ipv6.c - prosty program demonstrujący adresację IPv6 (wraz z konwersja postaci napisowej na numeryczną) obliczający adres sieci na podstawie maski i adresu hosta
— nazwy_hostow.c - prosty program demonstrujący przekształcanie nazwy hosta na adres IP
— UDP_nadajnik.c, UDP_odbiornik.c, TCP_serwer.c - zestaw programów pokazujących komunikację przez UDP i TCP
— jabber.cpp - prosta komunikacja przez sieć, z wykorzystaniem protokołu Jabber i napisów języka C++

numeracja_ipv6.c

```
/**
  * program oblicza adres sieci
  * na podstawie podanego adresu IPv6 hosta z tej sieci i maski
**/

#include <stdio.h>
#include <errno.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/types.h>

main (int argc, char *argv[], char *envp[]) {
 if (argc != 3){
  puts("zla liczba argumentow");
  return -1;
 }

// konwertujemy adres IPv6 z napisu na strukture, zawierajaca tablice 16 bajtow
 // odpowiadajacyh kolejnym bajtom adresu
 struct in6_addr adres;
 if (inet_pton(AF_INET6, argv[1], &adres) <= 0 ) {
  if (errno == 0)
   puts( "podany adres jest nieprawidlowy");
   // inet_pton nie ustawia wtedy errno ale zwrac 0 ...
  else
   perror( "próba zamiany adresu nieudana" );
  return -2;
 }

// tworzymy analogiczna strukture z tablica odpowiadajaca bajtom maski
 int msk = atoi(argv[2]);
 struct in6_addr maska;

int i;
 for (i=0; i<16; i++){

  if (i<msk/8)  // (1)
   maska.s6_addr[i]=0xff;
  else if (i==msk/8)  // (2)
   maska.s6_addr[i]=(0xff << (8-msk%8));
  else // (3)
   maska.s6_addr[i]=0;
 }

// wykonujemy and binarny miedzy adresem ip hosta a maska otrzymujac adres sieci
 // w zasadzie mozna by to zrobic w powyzeszej petli - zamiast tworzyc pelny opis maski
```

```
 // dla (1) nic nie robic, dla (2) wykonac and dla tego bajtu dla (3) wyzerowac bajty adresu
 for (i=0; i<16; i++)
  adres.s6_addr[i] = maska.s6_addr[i] & adres.s6_addr[i];

// wypisujemy (dokonujac konwersji na postac napisowa) otrzymany adres sieci
 char dst[40];
 if (NULL == inet_ntop  (AF_INET6, &adres, dst, 40)) {
  perror( "próba zamiany adresu nieudana" );
  return -3;
 }
 printf("%s\n", dst);
}
```

NAZWY_HOSTOW.C

```
#include <netdb.h>
#include <sys/socket.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>

main(int argc, char **argv) {
 int i=0;
 char buff[40];
 void* adres;

// zamiana nazwy na adresy
 struct hostent *info1 = gethostbyname(argv[1]);
 if (info1 == NULL) {
  herror("gethostbyname: ");
  return -1;
 }

struct in_addr adres_v4;
 struct in6_addr adres_v6;
 // opisy in6_addr i sockaddr_in6 w man 7 ipv6
 // opisy in_addr i sockaddr_in w man 7 ip

// wypisanie uzyskanych informacji
 printf("%s MA ADRESY (%d %d):\n", info1->h_name, info1->h_addrtype, info1->h_length);
 while (info1->h_addr_list[i]) {
  if (info1->h_addrtype == AF_INET) {
   adres = &adres_v4;
   memcpy ((void *) &adres_v4.s_addr, (void*)info1->h_addr_list[i], info1->h_length);
   printf("\tadres ip w postaci HEX: %x (%x)\n", adres_v4.s_addr, *((int*)info1->h_addr_list[
   i]));
  } else if (info1->h_addrtype == AF_INET6) {
   adres = &adres_v6;
   memcpy ((void *) &adres_v6.s6_addr, (void*)info1->h_addr_list[i], info1->h_length);
   printf("\tadres ip w postaci HEX: ");
   for (i=0; i<16; i++)
    printf("%x ", adres_v6.s6_addr[i]);
   printf("\n");
  }

  // konwersja wartości numerycznej na ludzki zapis adresu
  // funkcja inet_ntop rozpoznaje przyjmowany typ struktury
  // mogą to być n6_addr, sockaddr_in6, in_addr, sockaddr_in
```

```
  if (!inet_ntop  (info1->h_addrtype, adres, buff, 40)) {
   perror( "próba zamiany adresu nieudana" );
   return -1;
  }
  printf("\t%s %s\n\n", info1->h_aliases[i], buff);
  i++;
 }



struct addrinfo *info2;
 errno = 0;
 int error = getaddrinfo(argv[1], NULL, NULL, &info2);
 if (error != 0) {
  perror("getaddrinfo: ");
  printf("getaddrinfo: %d %s\n", error, gai_strerror(error));
 }

struct sockaddr_in *socket_v4;
 struct sockaddr_in6 *socket_v6;

printf("Za pomocą getaddrinfo\n");
 do {
  // UIWAGA: to co siedzi w info2->ai_addr to sockaddr_in lub sockaddr_in6 a nie sockaddr !!!
  if (info2->ai_family == AF_INET) {
   socket_v4 = (const struct sockaddr_in*) info2->ai_addr;
   adres = &socket_v4->sin_addr;
   printf("\tadres ip w postaci HEX: %x\n", socket_v4->sin_addr.s_addr);
  } else if (info2->ai_family == AF_INET6) {
   socket_v6 = (const struct sockaddr_in6*) info2->ai_addr;
   adres = &socket_v6->sin6_addr;
   printf("\tadres ip w postaci HEX: ");
   for (i=0; i<16; i++)
    printf("%x ", socket_v6->sin6_addr.s6_addr[i]);
   printf("\n");
  }

  // konwersja wartości numerycznej na ludzki zapis adresu
  if (!inet_ntop  (info2->ai_family, adres, buff, 40)) {
   perror( "próba zamiany adresu nieudana" );
   return -1;
  }
  printf("\t%s %s (%d %d)\n\n", info2->ai_canonname, buff, info2->ai_family, info2->
  ai_addrlen);

  // bierzemy następny z uzyskanych adresów
  info2=info2->ai_next;
 } while(info2!=NULL);
}
```

UDP_NADAJNIK.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody komunikacji sieciowej (TCP/IP) - program wysyłający dane przez UDP
 *
 * wywolanie: ./program adres.ip port dane
 *
```

```
 */

#include <sys/socket.h>
#include <netinet/ip.h>

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>

#define BUF_SIZE 255

main(int argv, char *args[]) {
 int res;
 char buf[BUF_SIZE];

// gniazdo ...
 int sh = socket(PF_INET, SOCK_DGRAM, 0);
 if (sh<0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }

// wysylanie danych
 struct sockaddr_in to;
 to.sin_family=PF_INET;
 to.sin_port=htons(atoi(args[2]));
 inet_aton(args[1], &to.sin_addr);
 res=sendto(sh, args[3], strlen(args[3]), 0, (struct sockaddr *) &to, sizeof(struct ⤸
 sockaddr_in));

printf("%d\n", res);

// zamkniecie gniazda
 close(sh);
}
```

UDP_odbiornik.c

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody komunikacji sieciowej (TCP/IP) - program odbierający dane przez UDP
 *
 */

#include <sys/socket.h>
#include <netinet/ip.h>

#include <stdio.h>
#include <stdlib.h>
#include <errno.h>

#define _LOCAL_PORT_ 4033
#define _BUF_SIZE_ 255

main() {
 int res;
 char buf[_BUF_SIZE_];
```

```
// gniazdo ...
 int sh = socket(PF_INET, SOCK_DGRAM, 0);
 if (sh<0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }

// przypisanie adresu ...
 struct sockaddr_in addr;
 addr.sin_family=PF_INET;
 addr.sin_port=htons(_LOCAL_PORT_);
  // host to network short (porzadek bajtow) ... sa tez dla long i odwrotne ...
 addr.sin_addr.s_addr=INADDR_ANY;

res=bind(sh, (struct sockaddr *) &addr, sizeof(struct sockaddr_in));

// odbior danych ...
 struct sockaddr_in from;
 socklen_t fromlen;
 res=recvfrom(sh, buf, _BUF_SIZE_, 0, (struct sockaddr *) &from, &fromlen);

printf("%s\n", buf);

// zamkniecie gniazda
 close(sh);
}
```

TCP_serwer.c

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody komunikacji sieciowej (TCP/IP) - prosty serwer TCP echo
 *
 * przykładowy klijent TCP zademonstrowany jest w jabber.cpp
 *
 */

#include <sys/socket.h>
#include <netinet/ip.h>

#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <inttypes.h>

#define BUF_SIZE 255
#define QUERY_SIZE 1
#define PORT 4080


char buf[BUF_SIZE];

int main(int argv, char *args[]) {
 int res;

// gniazdo ...
 int sh = socket(PF_INET, SOCK_STREAM, 0);
```

```
 if (sh<0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }

// utworzenie struktury opisującej adres
 struct sockaddr_in serwer;
 serwer.sin_family=AF_INET;
 serwer.sin_port=htons(PORT);
  // host to network short (porzadek bajtow) ... sa tez dla long i odwrotne ...
 serwer.sin_addr.s_addr=INADDR_ANY;
  // INADDR_ANY oznacza że nasłuchujemy na każdym adresie IP danego hosta
  // (zamiast tego można określić konkretny adres)

// przypisanie adresu ...
 if (bind(sh, (struct sockaddr *) &serwer, sizeof(struct sockaddr_in)) < 0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }

while(1) {
  // otwarcie portu do nasluchiwania
  if (listen(sh, QUERY_SIZE) < 0) {
   fprintf(stderr, "%s\n", strerror(errno));
   exit(EXIT_FAILURE);
  }

  // odebranie połączenia
  struct sockaddr_in from;
  socklen_t fromlen=sizeof(struct sockaddr_in);
  int sh2 = accept(sh, (struct sockaddr *) &from, &fromlen);

  // tutaj moglibysmy utworzyc potomka zajmujacego sie tym polaczeniem ...
  // wtedy w macierzystym robimy close(sh2) a w potomnym close(sh)

  // tu mozna sprawdzic czy klijent uprawniony jak nie to close(sh2)

  FILE * net;
  net=fdopen(sh2, "r+");
  while(fgets(buf, BUF_SIZE, net))
   fputs(buf, net);
  fclose(net);
 }

// zamkniecie gniazda
 close(sh);
}
```

JABBER.CPP

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody komunikacji sieciowej (TCP/IP)
 *
 * jest to bardzo prosty program który komunikuje się z serwerem Jabber
 * (więcej info o Jabber'ze: www.jabber.org i www.jabberpl.org)
 * zasadniczo powstał on w celu wysyłania powiadomień o nowej poczcie
 * przy takim wykorzystaniu oczywiście fragment odpowiedzialny za odbiór odpowiedzi
```

```
 * jest bezurzyteczny, ale został zamieszczony w celach edukacyjnych
 *
 */

#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/ip.h>

#include <unistd.h>

#include <string>
#include <sstream>

#include <stdio.h>

#define _NUMER_PORTU_ 5222
#define _ADRES_SERWERA_ "jabber.domena"
#define _MAIL_USER_ "uzytkownik"
#define _MAIL_USER_PASSWORD_ "secret1"

int main(int argv, char *args[]) {
  // pierwszy parametr - JID
  // drugi parametr - wiadomość

// otwieramy gniazdo
 int sockfd = socket (PF_INET, SOCK_STREAM, 0);
 struct sockaddr_in adres;
  adres.sin_family = AF_INET;
  adres.sin_port = htons (_NUMER_PORTU_);
 struct hostent *serwer = gethostbyname (_ADRES_SERWERA_); // rozwiązujemy adres serwera
  memcpy ((void *) &adres.sin_addr.s_addr, (void*) serwer->h_addr, serwer->h_length);

// łączymy z serwerem
 connect (sockfd, (struct sockaddr *) &adres, sizeof (adres));

// tworzymy wiadomosc
 for (unsigned int i = 0; i<strlen(args[2]); i++) {
  // w tej pętli zamienimy znaczki < i > na inne gdyz ich wystepownie w XML'u nie powinno ↩
  mieć miejsca
  if (args[2][i] == '<') args[2][i] = '[';
  if (args[2][i] == '>') args[2][i] = ']';
 }
 std::ostringstream message; // to wygodny sposób obsługi napisów

message << "<stream:stream to='" << _ADRES_SERWERA_;
 message << "' xmlns='jabber:client' xmlns:stream='http://etherx.jabber.org/streams'>";
 message << "<iq id='auth' type='set'><query xmlns='jabber:iq:auth'><username>";
 message << _MAIL_USER_ << "</username>" << "<password>" << _MAIL_USER_PASSWORD_;
 message << "</password><resource>system</resource></query></iq>";
 message << "<message to='" << args[1] << "'><body>" << args[2] << "</body></message>";
 message.put(0);
 //printf("%s", (message.str()).c_str());

// wysylamy dane
 send(sockfd, (message.str()).c_str(), (message.str()).length()-1, 0);


// ODBIOR DANYCH
 // bardzo prosta funkcja odbioru danych - mozna by ją wykorzystać do kontroli poprawności ↩
```

```
wysłania wiadomości ...
 struct timeval czas_1,czas_2;
 fd_set zestaw_1,zestaw_2;
 czas_1.tv_sec = 4; // limit czasu oczekiwania na dane - czas_1.tv_usec dla limitu w ⤸
milisekundach
 FD_ZERO(&zestaw_1);
 FD_SET(sockfd, &zestaw_1);
 bool quit = false;

while (quit == false) {
  czas_2 = czas_1;
  zestaw_2 = zestaw_1;
  select (FD_SETSIZE, &zestaw_2, 0, 0, &czas_2); // czekamy na dane w deskryptorach gniazd
  if (FD_ISSET (sockfd, &zestaw_2)) // serwer coś nadał
  {
   char bufor[1000];
   int bajty = read (sockfd, bufor, 1000);
   if (bajty == 0) // zostaliśmy rozłączeni
   {
    close (sockfd);
    FD_CLR (sockfd, &zestaw_1);
    quit = true;
   }
   printf("ODPOWIEDZ: %s\n", bufor);
   quit = true;
  }
 }


// wychodzimy
 close(sockfd);

return 0;
}
```

Rzeczy trochę ciekawsze

- alokacja_pamieci.c - ręczna alokacja pamięci oraz mapowanie plików na obszary pamięci (zobacz też: w wikibooks[998])
- openmp.c - urównoleglanie obliczeń z wykorzystaniem OpenMP[999], zobacz też Kurs OpenMP - pierwsze kroki[1000]
- c_lua_api.c - wykorzystanie języka skryptowego Lua w programach w C, z programem współpracuje skrypt c_lua_api.lua
- wyrazenia_regularne.cpp - wyrażenia regularne w C++
- watki.cpp - wątki, zdarzenia asynchroniczne, zagadnienia czasowe, ... w C++

Warto też zajrzeć na C++0x[1001] aby zobaczyć jakie zmiany planowane są w następnej wersji standardu C++.

alokacja_pamieci.c

```
/*
 * plik ma na celu prezentację metod ręcznego zarządzania pamięcią
 * oraz możliwości korzystania z plików mapownych na obszary pamięci
 *
 */


#include <stdio.h>
```

```c
#include <stdlib.h> /* for malloc itp */
#include <string.h> /* for memcpy */
#include <fcntl.h> /* for open */
#include <sys/mman.h> /* for mmap */

int main (int argc, char *argv[]) {
  /// ręczna alokacja pamięci
  /// na ogół podobny efekt można uzyskać deklarując tablica[jakas_zmienna]
  /// ale to nie działa w starszych implementacjach C
  /// zobacz: http://pl.wikibooks.org/wiki/C/Wskaźniki#Dynamiczna_alokacja_pami.C4.99ci
  /// ponadto metoda tablica[wartosc] narzuca ograniczenie na ich rozmiar związane
  /// z ograniczeniem rozmiaru stosu, dlatego lepiej duże tablice alokować dynamicznie


// wskaźnik na naszą tablicę
 char * tab = NULL;
 // rozmiar naszej tablicy
 int tab_size = 15;
 // aloakcja pamięci dla naszej tablicy + nadanie poprawnej wartości wskażźnikowi na nią
 // podobnie działa calloc z tym że dodatkowo zarezerwowany obszar zapisuje zerami
 tab = malloc(sizeof(*tab) * tab_size );

/// wppisanie czegoś do naszej tablicy
 memcpy(tab, "Ala ma kota", 12); // długość napisu + 1 bo null-end string

// sprawdzenie
 printf("%s - %c\n", tab, tab[4]);

// ustalenie nowego rozmiaru i inne pomocnicze
 // aby zobaczyć że następuje przeniesienie należy program wywołać z dwoma argumentem
 // stanowić on będzie rozmiar nowej tablicy oraz zpowoduje zajęcie pamięci za naszą ⤸
 tablicą
 // użyicie rozmiaru
 if (argc > 1) {
  tab_size = atoi(argv[1]);
  char *tab2 = malloc(sizeof(*tab) * 1000 );
  printf("%s\n\n", tab2);
 } else {
  tab_size = 40;
 }
 char *old_tab = tab;

/// zwiększenie rozmiaru naszej tablicy
 // jeżeli nie jest możliwe zwiększenie obecnego rozmiaru (musi to być obszar ciągły)
 // alokowany jest nowy obszar, a dane są kopiowane
 // w związku z powyższym zwrócony wskaźnik może się różnić od podanego !!!
 tab = realloc(tab, sizeof(*tab) * tab_size);

if (old_tab == tab)
  puts("nieprzeniesione");
 else
  puts("przeniesione");

// sprawdzenie
 printf("%s - %c\n", tab, tab[25]);
 memcpy(tab, "Ala ma kota. Kot ma Ale. Czyzby ...", tab_size);
 printf("%s - %c\n", tab, tab[25]);

/// usuwamy naszą tablice
 free(tab); // zwalniamy ten obszar pamięci
```

```
 tab = NULL; // kasuję wartośc wskaźnika (bo obszar zwolniony)

/// zamapowanie fragmentu pliku (lub urządzenia) do pamięci
 int deskryptor = open( "/etc/passwd", O_RDONLY ); // zobacz unix.c

tab = mmap(NULL, 50, PROT_READ, MAP_SHARED, deskryptor, 0);
 // mapujemy 50 bajtów z przesunięciem 0 (musi być ono być wielorotnośią
 // wartości PAGESIZE - zobacz getpagesize()) z pliku określonego przez
 // deskryptor na wspólny dla wszystkich kożytsających z pliku (MAP_SHARED)
 // obszar pamięci do odczytu w przypadku zapisów ważne jest wykonanie
 // msync (synchronizacji)lub munmap (odmapowania); ilość wczytanych danych
 // jest zaokrąglana w górę do całkowitej wielokrotności PAGESIZE
 //
 // właśnie w przypadku MAP_SHARED jest to wygodny mechanizm współdzielenia pamięci
 // przez kilka procesów, jeżeli chemy mieć włany bufor należy skorzystać z MAP_PRIVATE

// sprawdzenie
 printf("%s\n", tab);

/// odmapowanie
 munmap(tab, 20);
 tab=NULL;

// na koniec warto wspomniec takze o mozliwosci zablokowania wyswapowywania
 // pamięci naszego procesu lub jej części przy pomocy funkcji mlock lub mlockall
}
```

OPENMP.C

```
#include <omp.h>
#include <stdio.h>

// kompilacja: gcc -fopenmp openmp.c -o openmp_demo
// uruchomienie (4 wątki): env OMP_NUM_THREADS=4 ./openmp_demo
int main (int argc, char *argv[]) {
 int ilosc_watkow = -22, numer_watka = -33;
 // od tego miejsca kod bedzie zrownoleglany ...
 // default(none) wymusza jawne deklarowanie zasięgu każdej zmiennej
 // zmienna prywatna dla kazdego z watkow bedzie numer_watka
 // wspólną zmienną będzie ilosc_watkow
 #pragma omp parallel default(none) shared(ilosc_watkow) private(numer_watka)
 // ponadto w dyrektywie parallel możemy:
 // określić kiedy ma mieć miejsce zrównoleglenie - poprzez if (warnunek na jakis ⤸
zmiennych)
 // ile wątków powinno obsługiwać urównoleglenie - poprzez num_threads(ilosc)
 {
  // poniższy blok wykona się tylko w pojedyńczym wątku
  #pragma omp single
  {
   ilosc_watkow = omp_get_num_threads();
   printf("Urównoleglilişmy się na %d watków\n", ilosc_watkow);
  }

  // poniższy kod wykona każdy z wątków niezależnie
  numer_watka = omp_get_thread_num();
  printf("Tu wątek numer %d ... ja żyje :-)\n", numer_watka);

  // czekamy az wszystkie watki zakoncza powyzszy kod
```

```
  #pragma omp barrier
  if ( numer_watka == 0 ) { // tylko w watku zerowym
   printf("Tu wątek numer zero ... informuje iż każdy wątek ma już ustawioną swoją ⤸
   zmienną numer_watka\n");
  }

  // możemy też w prostrzy od pokazanego powyżej if'a przydzielać zadania poszczególnym ⤸
  wątkom:
  #pragma omp sections
  {
   #pragma omp section
   {
    // ten wątek sobie troszkę pośpi
    sleep(1);
    printf("Jakiś wątek AAAA\n");
   }
   #pragma omp section
   {
    printf("Jakiś inny wątek BBBB\n");
   }
  }
 }
 printf("No i nie ma już naszych %d wątków ... ", ilosc_watkow);
 printf("teraz mamy zwykły sekwencyjnie wykonywany program, a numerem wątka jest %d\n", ⤸
 numer_watka);
 // na powyższym widać różnicę między zmienną współdzieloną a prywatną dla ⤸
 wątków

// możemy urównoleglać także petlę for
 // możemy używać zmiennej do której pisac będzie kilka wątków
 // pod warunkiem że będzie to tablica a watki piszą do różnych komórek
 // lub że będzie to zmienna akumulacyjna (jak w poniższym przykładzie)
 // - musi być ona wtedy podana przez reduction(charakter:zmienna)
 int i, max_i=10, suma=0;
 #pragma omp parallel for default(none) shared(max_i) private(i) reduction(+ : suma)
 for (i = 0; i < max_i; i++) {
  suma += i * omp_get_thread_num();
 }
 printf("SUMA = %d\n", suma);
}
```

C_LUA_API.C

```
/*
 * Plik prezentuje korzystanie z skryptów Lua w środowisku C
 * wraz z tym kodem udostępniany jest także c_lua_api.lua
 * który jest przykładem skryptu Lua stworzonym do współpracy z tym programem
 *
 * kompilacja: gcc my.c -llua5.1
 *
 */

#include <lua5.1/lua.h>
#include <lua5.1/lauxlib.h>
#include "lgencall.h"

static int funkcja_dla_lua (lua_State *LS) {
 // zdejmujemy ze stosu liczbę argumentów przekazaną do funkcji
```

```
 int liczba_argumentow = lua_gettop(LS);
 int i;
 lua_Number wynik = 0;
 for (i = 1; i <= liczba_argumentow; i++) {
  int arg_type = lua_type(LS, i);

  // wypisujemy każdy argument
  printf("Agrument nr %d typu %s (%d) wartosc ", i, lua_typename(LS, arg_type), arg_type);
  switch (arg_type) {
   case LUA_TNUMBER:
    printf("%f\n", lua_tonumber(LS, i));
    break;
   case LUA_TSTRING:
    printf("%s\n", lua_tostring(LS, i));
    break;
   default :
    printf("(nieobslugiwany typ)\n");
  }

  // jeżeli argument jest liczbą to sumujemy
  // w zasadzie moznaby to robic w powyzszym case, ale chialem pokazac funkcje lua_isnumber
  if (lua_isnumber(LS, i))
   wynik += lua_tonumber(LS, i);
 }
 printf("Wynik %f\n", wynik);

// odkladamy na stos wartosci zwracane przez funkce
 lua_pushnumber(LS, wynik);
 lua_pushstring(LS, "UDALO SIE");

// zwracamy liczbę argumentow odlozonych jako wynik
 return 2;
}

main(int argc, char *argv[]) {
 if (argc <2) {
  puts("Nie podano argumentu określającego skrypt Lua");
   return -1;
 }
 // wywolujemy funckje tworzaca nowy "kontekst" obsługi skryptów Lua
 // w rzeczywistości jest to funkcja z lauxlib obudowująca wywaołania lua_newstate itd
 lua_State *LS = luaL_newstate();
 if (LS == NULL) {
  puts("Nie udało się zainicjalizować systemu Lua");
  return -1;
 }

// ładujemy do wskazanego kontekstu wszystkie biblioteki języka Lua
 luaL_openlibs(LS);

// korzystamy z kolejnej funkcji lauxlib
 // tym razem upraszcza ona ładowanie skryptów do "kontekstu"
 if ( luaL_loadfile(LS, argv[1]) ) {
  // wyświetlamy błąd zwrócony przez lua - jest na szczycie stosu
  printf("Nie udało się załadować skrypt Lua %s", lua_tostring(LS, -1));
  // kończymy działanie
  lua_close(LS);
  return -1;
 }
```

```
// przygotowujemy dane na których będzie operował skrypt - tablica
 lua_newtable(LS);

lua_pushnumber(LS, 1); // kładziemy indeks
 lua_pushfstring (LS, "a=%d b=%s", 3, "aby"); // kładziemy wartość
 lua_rawset(LS, -3); // tworzymy parę zgrupowaną w tablicy

lua_pushstring(LS, "dwa"); // powtarzamy dla kolejnych elementów tablicy ...
 lua_pushnumber(LS, 8);
 lua_rawset(LS, -3);

lua_setglobal(LS, "tabelka"); // nadajemy nazwę naszej tablicy

// funkcja
 lua_pushcfunction(LS, funkcja_dla_lua);
 lua_setglobal(LS, "funkcja_z_c");

// wykonanie skryptu
 if ( lua_pcall(LS, 0, LUA_MULTRET, 0) ) {
  printf("Nie udało się wykonać skrypt Lua %s", lua_tostring(LS, -1));
  lua_close(LS);
  return -1;
 }

printf("Wynik %f %f %s\n",
  lua_tonumber(LS, -3), lua_tonumber(LS, -2), lua_tostring(LS, -1));
 // warto zwrócić uwagę na odwrotną kolejność odwołań niż kładzenia - stos ...
 // warto także zauważyć że lua_tonumber daje nam wartość zmiennoprzecinkową

// zdejmujemy wszystkie 3 wyniki ze stosu
 printf("Elementów na stosie: %d\n", lua_gettop(LS));
 lua_pop(LS, 3);

// uproszczona metoda wywołania wybranej funkccji skryptu lua
 // więcej na http://lua-users.org/wiki/LuaGenericCall
 int val;
 lua_genpcall(LS, "return jakas_funkcja(2)", ">%d", &val);
 printf("Zwrocilo: %d\n", val);

// niszczymy mechanizm obsługi Lua
 lua_close(LS);
}
```

WYRAZENIA_REGULARNE.CPP

```
/** plik prezentuje podstawy programowania obiektowego w C++ :
  *   korzystanie z wyrażeń regularnych
  *
  *   kompilacja: gcc -lboost_regex-mt wyrazenia_regularne.cpp
  **/

#include <boost/regex.hpp>
#include <iostream>
#include <string>

int main() {
 static const boost::regex wyrazenie1("(.*)(ko*t).*");
 static const boost::regex wyrazenie2("(ko*t)");
```

```
 static const boost::regex wyrazenie3("(ko.*?) ");
 // dodanie ? w tym wyrazeniu powoduje jego niezachlanne dzialanie
 // (dopasowuje najmniej jak sie da)
 static const boost::regex wyrazenie4(" +");

boost::match_results<std::string::const_iterator> dopasowania;
 std::string napis1 = "Ala ma kooooota";
 std::string napis2 = "Kot ma Ale ...";
 std::string napis3 = "kot psu kotem a kotowi psem";

/* sprawdzanie czy pasuje */
if (regex_match(napis1, dopasowania, wyrazenie1)) {
  std::cout << "napis1 pasuje" << std::endl;
  std::cout << " dopasowany napis to \"" << dopasowania[0];
  // w kolejnych elementach dopasowania znajdują się napisyy odpowiadające dopasowanym ⤸
  podwyrażenim
  std::cout << "\" a pierwsze podwyrazenie ma wartość \""  << dopasowania[1] << "\"" << std:⤸
  :endl;
 } else
  std::cout << "napis1 nie pasuje" << std::endl;

if (regex_match(napis2, wyrazenie1))
  std::cout << "napis2 pasuje" << std::endl;
 else
  std::cout << "napis2 nie pasuje" << std::endl;

/* wyszukiwanie */
std::string::const_iterator poczatek = napis3.begin();
 std::string::const_iterator koniec = napis3.end();
 while (boost::regex_search(poczatek, koniec, dopasowania, wyrazenie3)) {
  std::cout << "Znalazlem: " << dopasowania[1] << "X" << std::endl;
  // tutaj podobnie jak wczesniej w 0 mamy cale dopasowane wyrazenie
  // a w kolejnych [] odpowiednie podwyrazenia

  // ustawiamy poczatek szukania na jeden znak dalej niz koniec znaleznionego tekstu
  poczatek = dopasowania[0].second;
 }

// możemy także uzyć iteratorow sregx
 boost::sregex_iterator begin(napis3.begin(), napis3.end(), wyrazenie3);
 boost::sregex_iterator end;
 while (begin != end){
  dopasowania = *begin;
  std::cout << "Znalazlem: " << dopasowania[1] << " X " << (*begin)[1] << std::endl;
  begin++;
 }

/* zastępowywanie */
napis1=regex_replace(napis1, wyrazenie2, "psa i \\1");
 std::cout << "Teraz: " << napis1 << std::endl;

/* dzielenie */
boost::sregex_token_iterator i(napis3.begin(), napis3.end(), wyrazenie4, -1);
 boost::sregex_token_iterator j;
 while(i != j) {
  std::cout << "Wydzielilem: " << *i++ << std::endl;
 }

// w boost są także inne implementacje wyrażeń regularnych - w tym umożliwiające
 // tworzenie wyrażeń na etapie kompilacji oraz operowanie bardziej złożonymi gramatykami
```

```
}
```

WATKI.CPP

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C++
 * prezentuje:
 *    - tworzenie programów wielowątkowych
 *    - korzystanie z mutexów
 *    - korzystanie z asynchronicznego io jako timera
 *    - korzystanie z kolejki C++
 *    - podatwy obsługi czasu ...
 *
 * kompilacja: gcc -lboost_date_time-mt -lboost_system-mt -lboost_thread-mt watki.cpp
 */


#include <queue>

#include <boost/thread/mutex.hpp>
#include <boost/thread/thread.hpp>

#include <boost/asio.hpp>
#include <boost/date_time/posix_time/posix_time.hpp>

#include <string>
#include <iostream>

std::queue<std::string> kolejka;

boost::mutex mutex;

int globalna=13;

class Watki {
public:
 int parametr_watku;
 void operator()() {
  std::cout << "to wątek z parametrem " << parametr_watku;
  std::cout << " globalna = " << globalna << std::endl;
  boost::this_thread::sleep(boost::posix_time::seconds(5));
  std::cout << "a teraz (nadal) z parametrem " << parametr_watku;
  std::cout << " globalna = " << globalna << std::endl;


  for (int i=0; i<6; i++) {
   boost::this_thread::sleep(boost::posix_time::millisec(700));
   boost::mutex::scoped_lock scoped_lock(mutex);
   // powyższy mutex blokuje od teraz do końca bloku
   // warto też wspomnieć o muteksach odpornych na blokowanie
   // już zablokowanego przez nas samych - boost::recursive_mutex

   if (!kolejka.empty()) {
    std::cout << i << ") odebrałem z kolejki " << kolejka.front() << std::endl;
    kolejka.pop();
   }
  }
```

```
}
};

void zdarzenie(const boost::system::error_code&) {
 std::cout << "budzik start" << std::endl;
 boost::this_thread::sleep(boost::posix_time::seconds(5));
 std::cout << "budzik stop" << std::endl;
}


int main() {
 Watki funktor_watek1;

funktor_watek1.parametr_watku=56;
 boost::thread watek1(funktor_watek1);
 boost::this_thread::sleep(boost::posix_time::seconds(2));
 funktor_watek1.parametr_watku=78;
 globalna=58;

boost::this_thread::sleep(boost::posix_time::seconds(2));

mutex.lock();
 kolejka.push("Ala ma kota");
 kolejka.push("Kot ma Ale");
 mutex.unlock();

std::cout << "2 piersze elementy sa w kolejce" << std::endl;
 boost::this_thread::sleep(boost::posix_time::seconds(2));

mutex.lock();
 kolejka.push("Abecadlo");
 kolejka.push("z pieca spadlo");
 mutex.unlock();

std::cout << "2 kolejne elementy sa w kolejce" << std::endl;

// asynchroniczne IO jako alarm
 boost::asio::io_service ios;
 boost::asio::deadline_timer budzik(ios, boost::posix_time::seconds(1.5));
 budzik.async_wait(zdarzenie);
 std::cout << "czekamy na budzik" << std::endl;
 ios.run(); // uruchomienie pętli obsługi
 std::cout << "po budziku" << std::endl;
}
```

RADIOMETEOR.C

```
/*
 * Program sluzacy do automatycznej detekcji radiometerow z
 * zarejestrowanych nagran audio (w postaci CDDA lub RAW LE)
 *
 * program stworzony na potrzeby Koła Naukowego Astronomów przy Uniwersytecie Warszawskim
 * (http://www.astrouw.edu.pl/~nalezyty/knastr/dokonania/radiometeory/)
 *
 * wersja: beta 2005-08-11
 *
 */
```

```c
/*
 wav2cdr --inwav --tocdr $f > $f.cdr_in
 for f in *.cdr; do wav2cdr --towav --incdr $f > $f.wav; done;
*/


/*******************
 *                 *
 *   KONFIGURACJA   *
 *                 *
 *******************/
// konfiguracja parametrow porownania
#define _WAGA_POROWNANIA_ 1.45

/** gdy dane na wejsciu jest mono zakomentowac zdefiniowanie _STEREO_ **/
#define _STEREO_ 1
/** gdy stereo odkomentowac ponizsze aby brac PRAWY kanal (standardowo brany LEWY) **/
//#define _RIGHT_ 1

/** odkomentowac gdy progrm ma zapisywac probki **/
#define _SAVE_AUDIO_

/** odkomentowac gdy progrm ma sprawdzac po przesunieciu o 1/2 _OKRES_PROBKOWANIA_ **/
/** pomocne gdy meteoryt ukryl sie na zmianie okresow probkowania **/
//#define _SEEK_

/** odkomentowac aby wprowadzic odrzucanie zbyt malych wartosci przy liczeniu sumy **/
//#define _MINIMALNA_ 2000

/** _OKRES_PROBKOWANIA_ * _MNOZNIK_CZASU_ = czestotliwosc propkowania pliku wejsciowego **/
/** _OKRES_PROBKOWANIA_ musi byc podzielny przez 2 bez reszty **/
#define _OKRES_PROBKOWANIA_ 4410
#define _MNOZNIK_CZASU_ 10
#define _MNOZNIK_SREDNIEJ_ 1



/***************
 *             *
 *   NAGLOWKI   *
 *             *
 ***************/
#include <stdio.h>
#include <math.h>



/***************
 *             *
 *   FUNKCJE   *
 *             *
 ***************/
int print_help() {
 printf("POMOC\n");

return 0;
}
```

```
/***************
 *              *
 *   PROGRAM    *
 *              *
 ***************/
int main(int argv, char *args[]) {

FILE *ifp, *ofp;

short nat=0, tmp1=0, tmp2=0, tmp3=0; // naderzenia dzwieku ...
 int suma=0, suma_sr=0;
 long poz=0, poz_tmp=0;
 long i=0, j=0, k=0, l=0, m=0, czas=0, last_czas=-1024; // liczniki
 int _SZUM_ = 14000000;

i = strlen(args[1]); char plik[i];
 i += 15; char nazwa_pliku_out[i]; i = 0;

// analiza argumentow i otwieranie pliku wejsciowego
 if ( argv == 2 ){
  if ( !strcasecmp(args[1],"--help") || !strcasecmp(args[1],"-h") ) {
   print_help();
   return 0;
  }

  sprintf(plik, "%s", args[1]);
  ifp = fopen(plik, "r");
  if (ifp == NULL) {
   fprintf( stderr, "\nBłąd otwarcia pliku wejściowego - %s\n\n" , args[1]);
   return -1;
  }
// } else if ( argv == 1 ) {
//   ifp = stdin;
 } else {
  print_help();
  return -1;
 }


/** czytanie pliku **/
#ifdef _SEEK_
#  ifdef _STEREO_
#    ifdef _RIGHT_
 fseek(ifp, _OKRES_PROBKOWANIA_ + 2, SEEK_CUR);
#    else
 fseek(ifp, _OKRES_PROBKOWANIA_, SEEK_CUR);
#    endif /* _RIGHT_ */
#  else
 fseek(ifp, _OKRES_PROBKOWANIA_ / 2, SEEK_CUR);
#  endif /* _STEREO_ */
#else
#  ifdef _STEREO_
#    ifdef _RIGHT_
 fseek(ifp, 2, SEEK_CUR);
#    endif /* _RIGHT_ */
#  endif /* _STEREO_ */
#endif /*   _SEEK_ */


while (fread(&tmp1, 1, 1, ifp) && fread(&tmp2, 1, 1, ifp)){
```

```
  /**
     LE -> BE
     w CDDA jest MSB,LSB (czyli low-endian)
     a procesor oczekuje LSB,MSB (czyli big-endian)
  **/
  nat = (tmp2 << 8) | tmp1;

  i++;
#ifdef _MINIMALNA_
  nat = abs(nat);
  if (nat > _MINIMALNA_)
   suma += nat;
#else
  suma += abs(nat);
#endif
  if (i == _OKRES_PROBKOWANIA_) {
   if (suma > _WAGA_POROWNANIA_ * _SZUM_) {// wykryto dzwiek
    j++;
   } else if (j>0) {// dzwiek zakonczyl sie
    j *= _OKRES_PROBKOWANIA_; // dlugosc w ms
    poz = ftell(ifp); // aktualna pozycja w pliku
    sprintf(nazwa_pliku_out, "%s_%d.cdr", plik, czas);

    /** WYPISZ: plik_IN czas[s] xxxx plik_OUT **/
    fprintf(stdout, "%s %f %d %s\n", plik, czas/60.0, k, nazwa_pliku_out);

#ifdef _SAVE_AUDIO_   /** ZAPIS AUDIO **/
    if (czas - last_czas > 10) {
     ofp = fopen(nazwa_pliku_out, "w");

     /* ustalenie dlugosci nagrania (j) i punktu startu (poz_tmp) */
     j = 2*_OKRES_PROBKOWANIA_*_MNOZNIK_CZASU_;
#ifdef _STEREO_
     j *= 2;
#endif /* _STEREO_ */
     poz_tmp = poz - j;
     if (poz_tmp<0) poz_tmp=0;
     j *= 2;

     /* cofamy do punktu startu przepisywania */
     fseek(ifp, poz_tmp, SEEK_SET);
     /*   przepisywanie */
     for (l=0; l<j; l++){
      fread(&tmp3, 2, 1, ifp);
      fwrite(&tmp3, 2, 1, ofp);
     }
     /* cofamy do konca detekcji */
     fseek(ifp, poz, SEEK_SET);
    } else {
     printf("  nie zapisuje\n");
    }
#endif /* _SAVE_AUDIO_ */
    j = 0;
    last_czas = czas;

   }

   i=0; k++; m++;
   suma_sr += suma;
   suma = 0 ;
```

```
    if ( m == _MNOZNIK_SREDNIEJ_ ) {
     _SZUM_ = suma_sr / _MNOZNIK_SREDNIEJ_;
     suma_sr = 0;
     m = 0;
//         printf("_SZUM_ = %i\n", _SZUM_);
    }

    if (k == _MNOZNIK_CZASU_ ) { // co 1 sekunde
     k=0; czas++;
     fprintf(stderr, ".");
    }
   }
#ifdef _STEREO_
  fseek(ifp, 2, SEEK_CUR);
#endif /* _STEREO_ */
 }

fprintf(stderr, "\n");
 /** zakonczenie programu **/
fclose (ifp);
 return 0;
}
```

Graficzny interfejs użytkownika (GUI)

GTK jest biblioteką do tworzenia GUI, stworzoną na potrzeby GIMPa oraz wykorzystywaną w projekcie GNOME. Napisana jest w sposób obiektowy w języku C. Inną biblioteką do tworzenia GUI (i nie tylko ma moduły obsługi baz danych, sieci, grafiki 3D - OpenGL, ...) jest **Qt**, w której powstaje KDE. Jest to w pełni obiektowa biblioteka C++.

- text_on_framebuffer.c - program prezentujący korzystanie z surowego framebuffer'a i biblioteki FreeType do renderingu tekstu
- dsp_gui_gtk.c - GUI napisane w GTK dla zamieszczonego powyżej dsp.c; prezentuje podstawy korzystania z GTK, program przechwytuje także sygnał o śmierci potomka w celu potwierdzenia tego zdarzenia; więcej o GTK: GTK+ By Example[1002]
- dsp_gui_qt.cpp, dsp_gui_qt.h, Makefile - GUI napisane w Qt3 dla zamieszczonego powyżej dsp.c; prezentuje podstawy korzystania z Qt oraz pisanie Makefile i podstawy programowanie obiektowego (deklaracja klasy ...); więcej o Qt: Qt @ Wikibooks[1003]
- Qt_okno_glowne.h - minimalistyczny plik nagłówkowy pozwalający na zbudowanie okna programu stworzonego w QtDesigner
- Program prezentujący timer, przetwarzanie plików xml, tabelki i drzewaka, menu kontekstowe, tworzenie subokna oraz wyświetlanie komunikatów:
  - timer_subwindow_xml.cpp - główny plik programu (funkcja main)
  - timer_window.cpp i timer_window.h - klasa głównego okna
  - config_window.cpp i config_window.h - klasa okna konfiguracji
  - Qt_timer_subwindow_xml.tgz[1004] - archiwum z całością prgramu (wraz z plikami .ui)
- Program prezentujący rysowanie, obsługę sieci (http, gniazda), obsługe wielowątkowości i singletonu, a także pasek stanu paski narzędzi oraz menu:
  - graphics_net_menu_toolbar.cpp - główny plik programu (funkcja main)
  - paint_window.cpp i paint_window.h - klasa okna głównego (rysowanie)
  - net_window.cpp i net_window.h - klasa okna komunikacji sieciowej
  - info_thread.cpp i info_thread.h - klasa sub wątka
  - singleton.h - klasa singletonu
  - Qt_graphics_net_menu_toolbar.tgz[1005] - archiwum z całością programu (wraz z plikami .ui)
- MyWindow.cpp, MyWindow.h, main.cpp, main.h - program demonstrujący tworzenie GUI dla BeOS R5 i pochodnych (całość wraz z plikiem projektu - gui_beos_demo.tgz[1006])

— ncurses_1.c - zaawansowane wyświetlanie na terminalu tekstowym z wykorzystaniem biblioteki termcap
— ncurses_2.c - zaawansowane wyświetlanie na terminalu tekstowym z wykorzystaniem biblioteki terminfo
— ncurses_3.c - zaawansowane wyświetlanie na terminalu tekstowym z wykorzystaniem biblioteki ncurses

TEXT_ON_FRAMEBUFFER.C

```
/* program prezentuje wykorzystanie linuxowego framebuffer'a i biblioteki FreeType */

#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <inttypes.h>
#include <sys/mman.h>
#include <string.h>
#include <linux/fb.h>

#include <ft2build.h>
#include FT_FREETYPE_H
#include FT_GLYPH_H

#define DEV_FILE "/dev/fb0"

int    fbFD;
char*  fbBUF;
struct fb_var_screeninfo vinfo;
struct fb_fix_screeninfo finfo;
int    fbFrameSize, fbPixelSize;

inline void setARGB(int idx, uint8_t a, uint8_t r, uint8_t g, uint8_t b) {
 char* point = fbBUF + idx;
 *(point)   = b;
 *(point+1) = g;
 *(point+2) = r;
 *(point+3) = a; // 0 == transparent
}

inline void drawRectangle(int startX, int startY, int stopX, int stopY) {
 int x, y, lineOffet, pixelOffset;

if (stopX > vinfo.xres || stopX == 0)
  stopX = vinfo.xres;
 if (stopY > vinfo.yres || stopY == 0)
  stopX = vinfo.yres;
 if (startX < 0)
  startX = 0;
 if (startY < 0)
  startY = 0;

lineOffet = (vinfo.yoffset + startY) * finfo.line_length;
 for (y = startY; y < stopY; y++) {
  pixelOffset = lineOffet + (vinfo.xoffset + startX) * fbPixelSize;;
  for (x = startX; x < stopX; x++) {
   setARGB(pixelOffset, 150, 200, 200, 0);
   pixelOffset += fbPixelSize;
  }
  lineOffet += finfo.line_length;
 }
}
```

```
int printCharBitmap(FT_GlyphSlot slot, int startX) {
 int fX, fY, fR, xOffset, lineOffet, pixelOffset;

xOffset   = startX + slot->bitmap_left;
 lineOffet = (vinfo.yoffset + 200 - slot->bitmap_top) * finfo.line_length;

for(fY = 0; fY < slot->bitmap.rows; fY++) {
  pixelOffset = lineOffet + (vinfo.xoffset + xOffset) * fbPixelSize;
  fR = fY * slot->bitmap.width;

  for (fX = 0; fX < slot->bitmap.width; fX++) {
   if (pixelOffset > fbFrameSize)
    return 0;
   setARGB(pixelOffset, slot->bitmap.buffer[fR + fX], 255, 0, 0);
   pixelOffset += fbPixelSize;
  }
  lineOffet += finfo.line_length;
 }

startX += slot->advance.x >> 6;
 return startX;
}

void renderText(char* text, FT_Face ftFace) {
 int          i, maxChar, xPos = 100;

maxChar = strlen( text );
 for (i=0; i<maxChar; i++) {
  if ( FT_Load_Char(ftFace, text[i], FT_LOAD_RENDER) ) {
   puts("Error in FT_Load_Char");
   continue;
  }
  xPos = printCharBitmap(ftFace->glyph, xPos);
  if (xPos == 0) break;
 }
}

int main(int argc, char** argv) {
 char*         fontName;
 FT_Library    ftLib;
 FT_Face       ftFace;

fontName = argv[1];

// FreeType open font

if ( FT_Init_FreeType(&ftLib) )
  puts("Error in FT_Init_FreeType");

if ( FT_New_Face(ftLib, fontName, 0, &ftFace) )
  puts("Error in FT_New_Face");

if ( FT_Set_Char_Size( ftFace, 0, 26 * 64, 128, 128 ) )
  puts("Error in FT_Set_Char_Size");


// open device and get info
 fbFD = open(DEV_FILE, O_RDWR);
 if (fbFD == -1) {
  perror("Open framebuffer device: " DEV_FILE " error");
```

```
  exit(1);
 }

if (ioctl(fbFD, FBIOGET_FSCREENINFO, &finfo) == -1) {
  perror("ioctl FBIOGET_FSCREENINFO error");
  exit(2);
 }

if (ioctl(fbFD, FBIOGET_VSCREENINFO, &vinfo) == -1) {
  perror("ioctl FBIOGET_VSCREENINFO error");
  exit(3);
 }

if (vinfo.bits_per_pixel != 32) {
  puts("We only support 32 bit per pixel");
  exit(4);
 }

// open memory buffer
 fbPixelSize = vinfo.bits_per_pixel / 8;
 fbFrameSize = vinfo.xres * vinfo.yres * fbPixelSize;
 fbBUF=(char*)mmap(0, fbFrameSize, PROT_WRITE, MAP_SHARED, fbFD, 0);

// write to frame buffer
 memset(fbBUF, 0, fbFrameSize);
 renderText(argv[2], ftFace);

FT_Done_Face(ftFace);
 FT_Done_FreeType(ftLib);

// close
 munmap(fbBUF, fbFrameSize);
 close(fbFD);
 return 0;
}
```

DSP_GUI_GTK.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje bardzo popularną bibliotekę tworzenia GUI, wykorzystywaną m.in. w GNOME - ⤸
 GTK
 *
 * jest jednocześnie użytecznym interfejsem dla programu dsp.c, który implementuje
 * software'ową pętlę pomiędzy wejściem liniowym a wyjściem głośnikowym (dla ⤸
 układów CMI9880)
 *
 *
 * kompilacja:
 *    gcc -Wall -g plik_zrodlowy.c -o plik_wynikowy `pkg-config --cflags gtk+-2.0` `pkg-config -⤸
 -libs gtk+-2.0`
 *
 * zachęcam do zapoznania się z poradnikiem na oficjalnej stronie projektu: http://www.gtk.⤸
 org/tutorial/
 * oraz polsko języcznym tekstem na 4programmers: http://4programmers.net/article.php?id=484
 * warto też zapoznać się z programem glade, który jest (przydatnym przy większych ⤸
 projektach
```

```
 * oraz do podejrzenia jaka funkcja wstawia to co nam akurat potrzeba) generatorem GUI w GTK
 *
 */

// nalezy ustawic sciezke i nazwe na ktora skompilowany jest dsp.c
#define DSP_PROG_PATH "/home/rrp/audio-video/tuner_tv_azx_unmute"
#define DSP_PROG_NAME "tuner_tv_azx_unmute"

// comenda ustawiajaca priorytety czasu rzeczywistego dla artsd oraz tuner_tv_azx_unmute
#define RT_PRI_PROG_CMD "sleep 1; PATH_TO_priorytet `pidof /usr/bin/artsd ⤸
PATH_TO_tuner_tv_azx_unmute`;"

#include <gtk/gtk.h> // nagłówki GTK

#include <unistd.h> // dla: fork(), execl(), execlp()
#include <stdlib.h> // dla: system()

#include <signal.h> // dla: kill(), signal()
#include <sys/wait.h> // dla: wait()


/** funkcje związane z zdarzeniami **/

void window_close(GtkWidget *widget, gpointer *data) {
 gtk_main_quit(); // zamyka program GTK
}

void button1_action() {
 // jeżeli nie zamierzamy korzystać z podawanych do funkcji argumentów można ich nie ⤸
 podawać
 if (fork() == 0) {
  execlp("xawtv", "xawtv", 0);
 }
}

void button2_action() {
 if (fork() == 0) {
  execlp("gamix", "gamix", 0);
 }
}

void button3_action() {
 if (fork() == 0) {
  execlp("konsole", "konsole", "-e", "radio", 0);
 }
}

int copy_dsp_work=0, pid=-1;
void button4_action() {
 // w zasadzie możnaby tu umieścić kod tego programu, ale wersja tekstowa jest przydatna ⤸
 na
 // terminalu (fbtv, radio) oraz wydaje mi się że zaciemniłoby to obraz tego programu
 if (copy_dsp_work == 0) {
  pid = fork();
  if (pid == 0) {
   execl(DSP_PROG_PATH, DSP_PROG_NAME, 0);
  } else if (pid != -1){
   copy_dsp_work = pid;
   system(RT_PRI_PROG_CMD);
  }
```

```
 } else {
  kill(copy_dsp_work, 15);
  copy_dsp_work = 0;
 }
}

/** funkcja obsługi sygnału **/

void end_of_child(int sig) {
 // potwierdzamy zakonczenie potomka
 waitpid(-1, NULL, WNOHANG);
}


/** funkcja główna **/
int main(int argc,char *argv[]) {
/** ustawiamy obsługę sygnału informującego o zakończeniu pracy potomka **/
signal(SIGCHLD, &end_of_child);

/** domyślnie wykonujemy start **/
button4_action();


/** deklaracje wstępne GTK **/
GtkWidget *window, *table, *button1, *button2, *button3, *button4;
  // utworzenie wskaznikow do okna i komponentów

gtk_init(&argc, &argv); //inicjalizacja gtk wraz z przekazaniem argumentów linii poleceń


/** okno i jego ustawienia **/
window = gtk_window_new(GTK_WINDOW_TOPLEVEL);
  // utworzenie standardowego okna
  // (np. GTK_WINDOW_POPUP - bez ramki i guzikow zamkniecia powiekszenia ...)

table = gtk_table_new (3, 4, 0);
  // utworzenie tabeli (3 wiersze 4 kolumny), która posłuży do rozmieszczania składników
  okna
 gtk_container_add(GTK_CONTAINER(window), table); // umieszczenie tabeli w oknie

gtk_signal_connect(GTK_OBJECT(window), "destroy", GTK_SIGNAL_FUNC(window_close), NULL);
  // rakacja na sygnał związany z zamknięciem okna, mozna to tez zapisać:
 // g_signal_connect(G_OBJECT(window), "destroy", G_CALLBACK(window_close), NULL);

// parametry okna:
 gtk_window_set_title(GTK_WINDOW(window), "/dev/dsp copy program");
 //gtk_window_set_position(GTK_WINDOW(window), GTK_WIN_POS_CENTER);
 gtk_window_set_default_size(GTK_WINDOW(window), 430, 110);


/** składniki okna **/
// przycisk 1
 button1=gtk_button_new_with_label("TV application"); // utworzenie przycisku
 gtk_signal_connect(GTK_OBJECT(button1), "clicked", GTK_SIGNAL_FUNC(button1_action), NULL);
  // funkcja wykonana po nacisnieciu przycisku
 gtk_table_attach(GTK_TABLE(table), button1, 1, 2, 1, 2, GTK_SHRINK, GTK_SHRINK,0,0);
  // umieszczenie przycisku w tabeli (znajduje sie w kolumnach od pierwszej do drugiej
  // i wierszu od pierwszego do drugiego)
 gtk_widget_show(button1); // pokazanie przycisku
```

```
// przycisk 2
 button2=gtk_button_new_with_label("Mixer application"); // utworzenie przycisku
 gtk_signal_connect(GTK_OBJECT(button2), "clicked", GTK_SIGNAL_FUNC(button2_action), NULL);
  // funkcja wykonana po nacisnieciu przycisku
 gtk_table_attach(GTK_TABLE(table), button2, 2, 3, 1, 2, GTK_SHRINK, GTK_SHRINK,0,0);
  // umieszczenie przycisku w tabeli (znajduje sie w kolumnach od drugiej do trzeciej
  // i wierszu od pierwszego do drugiego)
 gtk_widget_show(button2); // pokazanie przycisku

// przycisk 3
 button3=gtk_button_new_with_label("Radio application"); // utworzenie przycisku
 gtk_signal_connect(GTK_OBJECT(button3), "clicked", GTK_SIGNAL_FUNC(button3_action), NULL);
  // funkcja wykonana po nacisnieciu przycisku
 gtk_table_attach(GTK_TABLE(table), button3, 3, 4, 1, 2, GTK_SHRINK, GTK_SHRINK,0,0);
  // umieszczenie przycisku w tabeli (znajduje sie w kolumnach od trzeciej do czwartej
  // i wierszu od pierwszego do drugiego)
 gtk_widget_show(button3); // pokazanie przycisku

// przycisk 4 - przełączany (mający stan wciśnięty)
 button4=gtk_toggle_button_new_with_label("Start / Stop"); // utworzenie przycisku
 gtk_toggle_button_set_active(GTK_TOGGLE_BUTTON(button4), 1);
  // wciśnięcie przycisku, stan przycisku sprawdza z koleji funkcja:
  // gtk_toggle_button_get_active(GTK_TOGGLE_BUTTON(button4))
 gtk_signal_connect(GTK_OBJECT(button4), "toggled", GTK_SIGNAL_FUNC(button4_action), NULL);
  // funkcja wykonana po nacisnieciu przycisku (toggled działa tak samo jak clicked)
 gtk_table_attach(GTK_TABLE(table), button4, 1, 4, 2, 3, GTK_FILL, GTK_SHRINK,0,0);
  // umieszczenie przycisku w tabeli (znajduje sie w kolumnach od pierwszej do czwartej
  // i wierszu od drugiego do trzeciego)
 gtk_widget_show(button4); // pokazanie przycisku

// ustawienia odstępów
 gtk_table_set_row_spacing (GTK_TABLE(table), 0, 20); // po zerowym wierszu
 gtk_table_set_row_spacing (GTK_TABLE(table), 1, 10); // po pierwszym wierszu
 gtk_table_set_col_spacing (GTK_TABLE(table), 0, 45); // po zerowej kolumnie
 gtk_table_set_col_spacing (GTK_TABLE(table), 1, 15);
  // po pierwszej kolumnie (tej w której jest pierszy przycisk)
 gtk_table_set_col_spacing (GTK_TABLE(table), 2, 15); // po drugiej kolumnie

/** ustawienia końcowe **/
gtk_widget_show(table); // pokazanie tabeli
 gtk_widget_show(window); // pokazanie okna

gtk_main(); //Główna funckja programu GTK

g_print("Koniec ...\n"); // wypisujemy na terminal ...
 if (copy_dsp_work != 0) kill(copy_dsp_work, 15);

return 0;
}
```

Makefile

```
# więcej informacj w dsp_gui_qt.c

PLIK_WYNIKOWY=dsp_gui_qt
PLIK_ZRODLOWY=dsp_gui_qt
```

```
# domyslna akacja (all) wymagająca pliku .o oraz .mo (tworzonych przez poniższe akcje)
# wykonuje linkowanie plików *.o i *.mo oraz bibliotek (-lqt-mt) w program wynikowy
all: ${PLIK_ZRODLOWY}.o ${PLIK_ZRODLOWY}.mo
g++ *.o *.mo -o ${PLIK_WYNIKOWY} -lqt-mt


# kompilacja zwykłych źródeł C++
%.o: %.cpp
# $< oznacza plik wejsciowy (wymieniony powyżej jako wymagany (po dwukropku) %.cpp)
# $@ oznacza plik wynikowy (wymieniony powyżej przed dwukropkiem %.o
#    gdzie % oznacza dowlna zadana nazwe pliku
 g++ -c $< -o $@ -I/usr/include/qt3/


# kompilacja plików generowanych przez MOC
%.mo: %.h
moc $< -o $@.m.cc
 g++ -c $@.m.cc -o $@ -I/usr/include/qt3/
 rm -f $@.m.cc
```

DSP_GUI_QT.CPP

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje bardzo popularną bibliotekę tworzenia GUI, wykorzystywaną m.in. w KDE - Qt
 *
 * jest jednocześnie użytecznym interfejsem dla programu dsp.c, który implementuje
 * software'ową pętlę pomiędzy wejściem liniowym a wyjściem głośnikowym (dla ⤸
 układów CMI9880)
 *
 *
 * kompilacja (w Debianie 3.1): g++ plik_zrodlowy.cpp -o plik_wynikowy -I/usr/include/qt3/ -⤸
 lqt-mt
 * jednak gdy chcemy tworzyć własne funkcje obsługi sygnałów to musimy skorzystać też ⤸
 z programu "moc"
 * i wtedy lepszym rozwiązaniem wydaje się być załączony Makefile
 *
 * zachecam do zapoznania się z tutorialem oraz dokumentacją na: http://www.jtz.org.⤸
 pl/Inne/QT-Tutorial/
 * oraz tutorialem na: http://free.of.pl/q/qtmoux/
 * warto też zapoznaść się z środowiskiem "Qt designer" (designer),
 * przydatnym przy większych projektach oraz do odnalezienia tego co nam akurat potrzeba ...
 *
 */

// nalezy ustawic sciezke i nazwe na ktora skompilowany jest dsp.c
#define DSP_PROG_PATH "/home/rrp/audio-video/tuner_tv_azx_unmute"
#define DSP_PROG_NAME "tuner_tv_azx_unmute"

#include <unistd.h> // dla: fork(), execl(), execlp()
#include <signal.h> // dla: kill()

// nagłówki definicyjne klas Qt:
#include <qapplication.h>
#include <qpushbutton.h>
#include <qfont.h>
#include <qvbox.h>
#include <qhbox.h>
```

```
// nagłówek z deklaracją klasy naszego okna, ...
#include "dsp_gui_qt.h"

/** konstruktor klasy naszego okna (funkcja wywoływana przy tworzeniu obiektu tego typu) **/
MyWindow::MyWindow( QWidget *parent, const char *name ) : QVBox( parent, name )
{
 resize( 550, 80 ); // określamy rozmiar naszego pudełka

QHBox *box1 = new QHBox( this ), *box2 = new QHBox( this );
 // tworzymy dwa pudełka poziome będące dziećmi naszego pudełka pionowego box0

// przyciski
 QPushButton *button1 = new QPushButton( "TV application", box1 ); // tworzymy przycisk
 QPushButton *button2 = new QPushButton( "Mixer application", box1 ); // tworzymy przycisk
 QPushButton *button3 = new QPushButton( "Radio application", box1 ); // tworzymy przycisk
 QPushButton *button4 = new QPushButton( "Start / Stop", box2 ); // tworzymy przycisk
 button4->setFont( QFont( "Times", 18, QFont::Bold ) ); // zmieniamy czcionke przycisku

connect( button1, SIGNAL(clicked()), this, SLOT(button1_action()) );
 connect( button2, SIGNAL(clicked()), this, SLOT(button2_action()) );
 connect( button3, SIGNAL(clicked()), this, SLOT(button3_action()) );
 connect( button4, SIGNAL(clicked()), this, SLOT(button4_action()) );
}


/** klasa naszego okna - funkcje akcji **/
void MyWindow::button1_action() {
 // jeżeli nie zamierzamy korzystać z podawanych do funkcji argumentów można ich nie ↩
 podawać
 if (fork() == 0) {
  execlp("xawtv", "xawtv", 0);
 }
}
void MyWindow::button2_action() {
 if (fork() == 0) {
  execlp("gamix", "gamix", 0);
 }
}
void MyWindow::button3_action() {
 if (fork() == 0) {
  execlp("konsole", "konsole", "-e", "radio", 0);
 }
}
void MyWindow::button4_action() {
 // w zasadzie możnaby tu umieścić kod tego programu, ale wersja tekstowa jest przydatna ↩
 na
 // terminalu (fbtv, radio) oraz wydaje mi się że zaciemniłoby to obraz tego programu
 if (copy_dsp_work == 0) {
  pid = fork();
  if (pid == 0) {
   execl(DSP_PROG_PATH, DSP_PROG_NAME, 0);
  } else if (pid != -1){
   copy_dsp_work = pid;
  }
 } else {
  kill(copy_dsp_work, 15);
  copy_dsp_work = 0;
 }
}
```

```
/** funkcja główna programu **/
int main( int argc, char **argv )
{
 QApplication a( argc, argv ); // tworzymy obiekt zarządzający zasobami aplikacji

MyWindow w; // utworzenie obiektu z klasy moje okno
 w.copy_dsp_work=0; // ustwienia poczatkowe zmiennej

a.setMainWidget( &w ); // ustalamy że pudełko box0 będzie głównym oknem aplikacji
 w.show(); // pokazujemy nasze okno ...

return a.exec();
   // pętla główna programu i zwrócenie kodu zwróconego przez a.exec() do rodzica ...
}

/* gdybyśmy chcieli nie używać kasy okna funkcja ta wyglądłaby jak poniżej:
 int main( int argc, char **argv )
 {
   QApplication a( argc, argv );
     // tworzymy obiekt zarządzający zasobami aplikacji

   QVBox box0;
     // tworzymy pudełko pionowe (ustawia obiekty wkładane do niego jeden pod drugim)
   box0.resize( 300, 80 );
     // określamy rozmiar naszego pudełka

   QHBox box1( &box0 ), box2 ( &box0 );
   // tworzymy dwa pudełka poziome będące dziećmi naszego pudełka pionowego box0

   // przyciski
   QPushButton button1( "TV application", &box1 ); // tworzymy przycisk
   QPushButton button2( "Mixer application", &box1 ); // tworzymy przycisk
   QPushButton button3( "Radio application", &box1 ); // tworzymy przycisk
   QPushButton button4( "Start / Stop", &box2 ); // tworzymy przycisk
   button4.setFont( QFont( "Times", 18, QFont::Bold ) ); // zmieniamy czcionke przycisku

   // wywolania funkcji wiazacych akcje typu:
   //    QObject::connect( &quit, SIGNAL(clicked()), &a, SLOT(quit()) );

   a.setMainWidget( &box0 ); // ustalamy że pudełko box0 będzie głównym oknem aplikacji
   box0.show(); // pokazujemy nasze okno ...

   return a.exec();
     // pętla główna programu i zwrócenie kodu zwróconego przez a.exec() do rodzica ...
 }
*/
```

DSP_GUI_QT.H

```
/*
 *
 * więcej informacj w dsp_gui_qt.c
 *
 */

#include <qvbox.h>
```

```
/** deklarujemy klasę naszego okna, w oparciu o pudełko pionowe (ustawia obiekty wkładane ⤸
do niego jeden pod drugim) **/
class MyWindow : public QVBox
{
 Q_OBJECT
public:
  MyWindow( QWidget *parent=0, const char *name=0 );
  int copy_dsp_work;
 public slots:
  void button1_action();
  void button2_action();
  void button3_action();
  void button4_action();
 protected:
  int pid;
};
```

Qt_okno_glowne.h

```
/**
 * minimalny plik .h konieczny do uruchomiaenia okienka o nazwie OknoGlowne zapisanego w ⤸
 pliku okno_glowne.ui
 *
 * KOMPILACJA: qmake -project && qmake && make
 * POSPRZĄTANIE: make distclean && rm -f *.pro
 *
 * uwaga jezeli taki scalony plik nazwalibysmy .cpp a nie .h to qmake miałby problem z 'moc'
 *
 * w dalszych zastosowaniach warto podzielic ten plik na:
 * nagłówek klasy (okno_glowne.h), ciało kallsy (okno_glowne.cpp), funkcje main (main.cpp)
 * (w obu pozostałych zainkludować okno_glowne.h)
 **/

#include <QApplication>
#include <QTextCodec>
#include <QDialog>

#include "ui_okno_glowne.h"

class OknoGlowne: public QDialog {
Q_OBJECT
public:
   Ui::OknoGlowne interfejs;
   OknoGlowne(QWidget * parent, Qt::WFlags f);
   ~OknoGlowne(){};
};

OknoGlowne::OknoGlowne(QWidget * parent = 0, Qt::WFlags f = 0 ):QDialog(parent, f | Qt::Tool) ⤸
{
 interfejs.setupUi(this);
}

int main(int argc, char *argv[]) {
 QApplication aplikacja(argc, argv);

QTextCodec::setCodecForTr (QTextCodec::codecForName ("UTF-8"));
```

```
 QTextCodec::setCodecForCStrings (QTextCodec::codecForName ("UTF-8"));

OknoGlowne okienko(0, 0);

okienko.show();

return aplikacja.exec();
}
```

TIMER_WINDOW.H

```
#ifndef TIMER_WINDOW_H
#define TIMER_WINDOW_H

/**
 * plik nagłówkowy klasy TimerWindow związanej z okienkiem "TimerWindow"
 * opisanym w pliku timer_window.ui poprzez klasę "TimerWindow" w przestzeni nazw Ui
**/

// pliki nagłówkowe wykorzystywanych klas Qt
#include <QDialog>
#include <QMessageBox>

// timer
#include <QDateTime>
#include <QTimer>

// menu kontekstowe
#include <QMouseEvent>
#include <QMenu>
#include <QAction>

// plik nagłówkowy zawierający opis wyglądu okienka
// wygenerowany automatycznie w oparciu o timer_window.ui
#include "ui_timer_window.h"

// deklaracja klasy
class TimerWindow: public QDialog
{
Q_OBJECT
public:
   Ui::TimerWindow interfejs;
   TimerWindow(QWidget * parent = 0, Qt::WFlags f = 0 );
   ~TimerWindow(){};
   void mousePressEvent(QMouseEvent *e);
   void closeEvent(QCloseEvent *event);
 public slots:
   void TimerTick();
   void ShowConfig();
   void StopApp();
 private:
   // prawo zamknięcia okienka
   bool can_close;

   // zmienne obiektu timera i określenia czasu końca odliczania
   QTimer *timer;
   QDateTime deadline;
```

```
  // obsługa paska postępu
  char progress;
  bool wzrost;

  // akcje menu kontekstowego
  QAction *act_start, *act_stop, *act_config, *act_exit;
  QMenu *menu_podreczne;
};

#endif
```

TIMER_WINDOW.CPP

```
#include "timer_window.h"
#include "config_window.h"

// funkcja którą powiążemy z sygnałem generowanym przez upływ timera
void TimerWindow::TimerTick() {
 // aktualna data i czas
 QDateTime teraz = QDateTime::currentDateTime();

// aktualizacja wypisanego czasu
 interfejs.lab_sekundy->setText(QString::number(teraz.secsTo(deadline)));

// obsługa paska postępu
 if (wzrost)
  progress += 4;
 else
  progress -= 4;

if (progress >= 100) {
  interfejs.progressBar->setInvertedAppearance(true);
  wzrost=false;
 } else if (progress <= 0) {
  interfejs.progressBar->setInvertedAppearance(false);
  wzrost=true;
 }

interfejs.progressBar->setValue(progress);


// ponowne wystartowanie timera (zadziała za 200ms)
 timer->start( 100 );
}

// funkcja wyświetlająca okno konfiguracji
void TimerWindow::ShowConfig() {
 ConfigWindow okno_konfiguracji(0, 0);
 okno_konfiguracji.show();
 okno_konfiguracji.exec();
}

// funkcja (zdarzenie) reagująca na kliknięcie - wyświetla menu
void TimerWindow::mousePressEvent(QMouseEvent *e) {
 if (e->button() == Qt::RightButton) {
  if (timer->isActive()) {
   act_start->setEnabled(false);
```

```
   act_stop->setEnabled(true);
  } else {
   act_start->setEnabled(true);
   act_stop->setEnabled(false);
  }
  menu_podreczne->exec(e->globalPos());
 }
}

// funkcja (zdarzenie) reagująca na próbę zamknięcia okienka
void TimerWindow::closeEvent(QCloseEvent *event) {
 // jeżeli nie wyrażono zgody - brak potwierdzenia to nie zamykamy
 if (can_close) {
  event->accept();
 } else {
  if (QMessageBox::Yes == QMessageBox::question (this,
   "Potwierdzenie", "Czy napewno zakończyć program ?",
   QMessageBox::Yes | QMessageBox::No, QMessageBox::No)) {
    StopApp(); // w zasadzie wystarczyłoby tu qApp->quit();
    // użycie qApp->quit(); w tej funkcji jest konieczne aby zakończyć program
    // uruchomiony przez .exec() na obiekcie typu QApplication
    // jeżeli exec był na okienku to wystarcza standardowy close()
    // który się wykona bez korzystania z tej funkcji jako reakcja
    // na zamknięcie krzyżykiem (prawda dla Qt::WFlags = Qt::Tool)
  } else {
   event->ignore();
  }
 }
}

// funkcja zamykająca program z menu podręcznego
void TimerWindow::StopApp() {
 timer->stop();
 can_close = true;
 qApp->quit();
}

// konstruktor
TimerWindow::TimerWindow(QWidget * parent, Qt::WFlags f):QDialog(parent, f | Qt::Tool) {
 // skonfigurowanie interfeju w oparciu o obiekt klasy wygenerowanej na podstawie pliku .ui
 interfejs.setupUi(this);

// określenie czasu końca i jego wyświetlenie w Label
 deadline = QDateTime(QDate(2007, 12, 14), QTime(23, 59, 59));
 interfejs.lab_data->setText(deadline.toString("(d MMMM hh:mm)"));

// inicjalizacja zmiennych dla paska postępu
 progress=0;
 wzrost=true;

// obsługa startu timera + powiązanie sygnału emiotowanego przez timer z TimerTick()
 timer = new QTimer(this);
 connect( timer, SIGNAL(timeout()), this, SLOT(TimerTick()) );
 timer->start( 100 );
 /* w powyższym kodzie widzimy zastosowanie systemu sygnałów biblioteki Qt
    dokonane jest tutaj powiązanie sygnału timeout generowanego przez obiekt timer
    z slotem (funkcją) TimerTick bierzacego obiektu

    jest to pewnego rodzaju implementacja wzorca obserwatora,
    inne tego typu rozwiązanie znaleźć można w bibliotece boost */
```

```
// obsługa menu
 menu_podreczne = new QMenu(this);

act_start = new QAction ("Start", this);
 menu_podreczne->addAction (act_start);
 // sygnał przesyłany bezpośrednio do timera
 connect( act_start, SIGNAL(triggered()), timer, SLOT(start()) );

act_stop = new QAction ("Stop", this);
 menu_podreczne->addAction (act_stop);
 connect( act_stop, SIGNAL(triggered()), timer, SLOT(stop()) );

menu_podreczne->addSeparator();

act_config = new QAction ("Konfiguracja", this);
 menu_podreczne->addAction (act_config);
 connect( act_config, SIGNAL(triggered()), this, SLOT(ShowConfig()) );

menu_podreczne->addSeparator();
 act_exit = new QAction ("Wyjście", this);
 menu_podreczne->addAction (act_exit);
 connect( act_exit, SIGNAL(triggered()), this, SLOT(StopApp()) );

// nie pozwalamy na zamykanie
 can_close = false;
}
```

TIMER_SUBWINDOW_XML.CPP

```
/**
 * Program prezentujący podstawy programowania w Qt4 - prezentuje sposoby podstawy
 * interfejsu, komunikaty, okno zapisu pliku, tworzenie subokna,
 * obsługę plików xml, timera oraz tabelek i drzewek ...
 *
 * jest plik główny programu z funkcją main
 * poostałe elementy umieszczone są w plikach klas obsługujących poszczególne okienka
 *
 * KOMPILACJA: qmake -project && echo 'QT += xml' >> *.pro && qmake && make
 * POSPRZĄTANIE: make distclean && rm -f *.pro config.NEW.xml
 **/

// nagłówki Qt
#include <QApplication>
#include <QTextCodec>

// nagłowek klasy związanej z głównym okienkiem aplikacji
#include "timer_window.h"

int main(int argc, char *argv[]) {
 QApplication aplikacja(argc, argv);

// ustawiamy kodowanie dla wyświetlanych i dla przetwarzanych napisów
 QTextCodec::setCodecForTr (QTextCodec::codecForName ("UTF-8"));
 QTextCodec::setCodecForCStrings (QTextCodec::codecForName ("UTF-8"));

// tworzymy okienko w oparciu o klase TimerWindow
 TimerWindow okienko(0, 0);
```

```
// wyświelamy okienko
 okienko.show();

// uruchamiamy aplikację
 // -> będzie działała nawet po zamknięciu okienka
 // -> trzeba kończyć przez qApp->quit() a nie przez close() na okienku
 return aplikacja.exec();
}
```

CONFIG_WINDOW.H

```
#ifndef CONFIG_WINDOW_H
#define CONFIG_WINDOW_H

// pliki nagłówkowe wykorzystywanych klas Qt
#include <QDialog>
#include <QMessageBox>
#include <QTableWidgetItem>
#include <QStandardItemModel>
#include <QTextStream>

#include <QtXml/QDomDocument>
#include <QtXml/QDomElement>
#include <QtXml/QDomNode>
#include <QtXml/QDomText>
#include <QFile>
#include <QFileDialog>

// plik nagłówkowy zawierający opis wyglądu okienka
// wygenerowany automatycznie w oparciu o timer_window.ui
#include "ui_config_window.h"

// deklaracja klasy
class ConfigWindow: public QDialog
{
Q_OBJECT
private:
   int max_id;
   QStringList naglowek_tabelki;
 public:
   Ui::ConfigWindow interfejs;
   ConfigWindow(QWidget * parent = 0, Qt::WFlags f = 0 );
   ~ConfigWindow(){};
 public slots:
   void SaveAction();
   void AddTableRow();
   void DelTableRow();
   void IsTableCliced(int r, int c);
};

#endif
```

CONFIG_WINDOW.CPP

```
#include "config_window.h"
```

```
// funkcja zapisująca konfigurację i zamykająca to okienko
void ConfigWindow::SaveAction() {
 // deklaracje dotyczące XML
 QDomDocument dokument_xml;
 QDomElement wezel_glowny, rekord, pole;
 QDomText pole_val;

// element główny - korzeń
 wezel_glowny = dokument_xml.createElement( "baza" );
 dokument_xml.appendChild(wezel_glowny);

for (int i=0; i<interfejs.tableWidget->rowCount(); i++) {
  // tworzymy nowy rekord odpowiadający wierszowi tabeli
  rekord = dokument_xml.createElement( "termin" );
  // pobieramy ID
  QTableWidgetItem *komorka = interfejs.tableWidget->item(i,0);
  // i ustawiamy jako atrybut naszego rekordu
  rekord.setAttribute( "id", komorka->text() );
  // dodajemy rekord do bazy
  wezel_glowny.appendChild(rekord);

  for (int j=1; j<interfejs.tableWidget->columnCount(); j++) {
   // tworzymy pole o nazwie odpowiadającej nazwie kolumny
   pole = dokument_xml.createElement( naglowek_tabelki[j] );
   // dodajemy je do naszego rekordu
   rekord.appendChild(pole);

   // pobieramy wartość pola
   komorka = interfejs.tableWidget->item(i,j);

   // ustawiamy ją w XML
   if ( komorka != NULL  ) {
    pole_val = dokument_xml.createTextNode( komorka->text() );
    pole.appendChild(pole_val);
   }
  }
 }
 // oczywiście zamiast generować cały xml od nowa można by modyfikować stary
 // ale przy edycji z wykorzystaniem tabelki tak jest łatwiej
 // modyfikację wykonuje się w podobny sposób jak generowanie nowego
 // zobacz funkcje: insertBefore(), insertAfter(), replaceChild(), removeChild()
 // oraz (dla atrybutów): removeAttribute()

// wybór pliku do zapisania
 QString plik_do_zapisu = QFileDialog::getSaveFileName(this,\
  "Wybierz plik...", "./config.NEW.xml", "Dokumenty XML (*.xml)");

if (!plik_do_zapisu.isEmpty ()) {
  // zapisujemy do pliku
  QFile dokument( plik_do_zapisu );
  dokument.open( QIODevice::WriteOnly );
  QTextStream ts( &dokument );
  ts << dokument_xml.toString();

  // zamykamy okienko
  close();
 }
}

// funkcja dodająca wiersz do tabelki
```

```
void ConfigWindow::AddTableRow() {
 int wiersz = interfejs.tableWidget->rowCount();
 interfejs.tableWidget->setRowCount(wiersz+1);
 QTableWidgetItem *tmp_komorka = new QTableWidgetItem( QString::number(++max_id) );
 interfejs.tableWidget->setItem(wiersz, 0, tmp_komorka);
}

// funkcja usuwająca wiersz z tabelki
void ConfigWindow::DelTableRow() {
 interfejs.tableWidget->removeRow(interfejs.tableWidget->currentRow());
}

// funkcja reagujaca na kliknięcie komórki tabelki
void ConfigWindow::IsTableCliced(int r, int c) {
 printf("KLIKNIĘTO %d %d\n", r, c);
}

// konstruktor
ConfigWindow::ConfigWindow(QWidget * parent, Qt::WFlags f):QDialog(parent, f | Qt::Tool) {
 // skonfigurowanie interfeju w oparciu o obiekt klasy wygenerowanej na podstawie pliku .ui
 interfejs.setupUi(this);

// powiązanie guzika z akcją
 connect( interfejs.btn_save, SIGNAL(clicked()), this, SLOT(SaveAction()) );
 connect( interfejs.btn_table_add, SIGNAL(clicked()), this, SLOT(AddTableRow()) );
 connect( interfejs.btn_table_del, SIGNAL(clicked()), this, SLOT(DelTableRow()) );
 connect( interfejs.tableWidget, SIGNAL(cellClicked(int, int)), this, SLOT(IsTableCliced(int, ↩
 int)) );

// przygotowanie tabelki
 naglowek_tabelki << "Id" << "Rok" << "Miesiac" << "Dzien" << "Godzina" << "Minuty";
 interfejs.tableWidget->setColumnCount(6);
 interfejs.tableWidget->setHorizontalHeaderLabels( naglowek_tabelki );
 interfejs.tableWidget->hideColumn(0);
 int wiersz = 0;

// przygotowanie drzewka
 QStandardItemModel *terminy = new QStandardItemModel ();
 // wstawiamy węzeł który będzie rodzicem - tak aby było to drzewko
 QModelIndex rodzic = terminy->index(0, 0, rodzic);
 terminy->insertRows(0, 1, rodzic);
 terminy->insertColumns(0, 6, rodzic);
 terminy->setData( terminy->index(0, 0, rodzic), QVariant( "TERMINY" ) );
 rodzic = terminy->index(0, 0, rodzic);
 terminy->insertColumns(0, 6, rodzic);
 // wstawiamy nagłówki kolumn
 terminy->setHeaderData (0, Qt::Horizontal, "Rok");
 terminy->setHeaderData (1, Qt::Horizontal, "Miesiac");
 terminy->setHeaderData (2, Qt::Horizontal, "Dzien");
 terminy->setHeaderData (3, Qt::Horizontal, "Godzina");
 terminy->setHeaderData (4, Qt::Horizontal, "Minuty");
 terminy->setHeaderData (5, Qt::Horizontal, "Id");

// wczytanie pliku xml
 QDomDocument dokument_xml;
 QFile dokument( "config.xml" );
 dokument.open( QIODevice::ReadOnly );
 dokument_xml.setContent( &dokument );
 dokument.close();
```

```
// bierzemy uchwyt do głównego elementu w pliku (wszystkie inne są wewnętrz niego)
QDomNode wezel_glowny;
wezel_glowny = dokument_xml.documentElement();

// rekordy i pola
QDomNode rekord, pole;

// przetwarzamy kolejne rekordy
rekord = wezel_glowny.firstChild();
do {
 // dodajemy nowy wiersz do tabelki i drzewka
 interfejs.tableWidget->setRowCount(wiersz+1);
 terminy->insertRows(wiersz, 1, rodzic);
 printf("Dodalismy wiersz %d\n", wiersz);

 // tymczasowy element związane z rekodami/polami
 QDomElement tmp_elem;
 tmp_elem = rekord.toElement();

 // atrybut ID - zapamiętujemy maksymalne ID
 int id_num=tmp_elem.attribute("id").toInt();
 if (id_num > max_id) {
  max_id = id_num;
 }
 printf(" id = %d\n", id_num);

 // atrybut ID (numeryczny) w ukrytej kolumnie
 QTableWidgetItem *tmp_komorka = new QTableWidgetItem( QString::number(id_num) );
 interfejs.tableWidget->setItem(wiersz, 0, tmp_komorka);

 // atrybut ID w drzewku
 terminy->setData( terminy->index(wiersz, naglowek_tabelki.count()-1, rodzic), QVariant( ⤸
 QString::number(id_num) ) );

 for (char i=1; i<naglowek_tabelki.count(); i++) {
  // pole o nazwie określonej w nagłówku tabeli
  pole = rekord.namedItem(naglowek_tabelki[i]);
  tmp_elem = pole.toElement();
  // wypisujemy nazwę pola oraz jego zawartość jako napis C kodowany w UTF-8
  puts( ( " -> " + tmp_elem.tagName() + ": " + tmp_elem.text() ).toUtf8().constData() );

  // wstawiamy dane do tabelki
  QTableWidgetItem *tmp_komorka = new QTableWidgetItem(tmp_elem.text());
  interfejs.tableWidget->setItem(wiersz, i, tmp_komorka);

  // wstawiamy dane do drzewka
  terminy->setData( terminy->index(wiersz, i-1, rodzic), QVariant(tmp_elem.text()) );
 }
 wiersz++;

 // następny rekord (o dowolnej nazwie)
 rekord = rekord.nextSibling();
} while (!rekord.isNull());

// włączamy sortowanie tabelki
interfejs.tableWidget->setSortingEnabled(true);
interfejs.tableWidget->sortItems(1);

// przypisujemy model do drzewka
interfejs.treeView->setModel( terminy );
```

```
 interfejs.treeView->setColumnHidden(naglowek_tabelki.count()-1, true);
}
```

SINGLETON.H

```
#ifndef SINGLETON_H
#define SINGLETON_H

#include <QString>
#include <inttypes.h>

class Singleton {
 public:
  // tutaj umieszczamy zmienne które ma przechowywać nasza klasa
  uint16_t port;
  QString adres_ip;

  // publiczna statyczna funkcja
  static Singleton& use() {
   // funkcja tworzy instancję tej klasy ale tylko gdy jej nie było
   // w zasadzie można by trzymać wskaźnik, sprawdzać if'em czy nie NULL
   // i gdy null to tworzyć obiekt przez new i zwracać wskaźnik do niego,
   // ale po co skoro możemy skorzystać z mechanizmu lokalnych zmiennych statycznych,
   // które są tworzone przy pierwszym napotkaniu deklaracji, a potem deklaracja ta
   // jest ignorowana i tylko z nich korzystamy, ponadto ich destruktory są wywoływane
   // automatycznie przy kończeniu programu ...
   static Singleton singleton_use;
   return singleton_use;
  }
 private:
  Singleton(){};
  Singleton(const Singleton&){};
  // oba domyślne konstruktory muszą być prywatne ...
  // aby nie dało się inaczej utworzyć tego obiektu
};

#endif
```

PAINT_WINDOW.H

```
#ifndef PAINT_WINDOW_H
#define PAINT_WINDOW_H

/**
 * plik nagłówkowy klasy PaintWindow związanej z okienkiem "PaintWindow"
 * opisanym w pliku timer_window.ui poprzez klasę "PaintWindow" w przestzeni nazw Ui
**/

// pliki nagłówkowe wykorzystywanych klas Qt
#include <QDialog>
#include <QMouseEvent>

#include <QPen>
#include <QBrush>
#include <QColor>

#include <QPoint>
```

```
#include <QImage>
#include <QPainter>
#include <QPicture>

#include <QGraphicsItem>

#include <QFile>
#include <QtNetwork/QHttp>
#include <QMessageBox>

#include <QProcess>
#include <QToolButton>
#include <QToolBar>

// plik nagłówkowy zawierający opis wyglądu okienka
// wygenerowany automatycznie w oparciu o timer_window.ui
#include "ui_paint_window.h"

// plik nagłówkowy klasy subwątka ...
#include "info_thread.h"

// deklaracja klasy
class PaintWindow: public QMainWindow
{
Q_OBJECT
private:
   // obrazek, pędzle, rysowanie, ...
   QPicture *rysunek;
   QImage *obrazek;
   QBrush *pedzel;
   QPen *rysik;

   // pobieranie http
   QMessageBox *info;
   bool trawa_pobieranie;
   QFile *plik_obrazka;
   QHttp *http;

   // ping + pasek statutu i narzędzi
   QProcess *pingProces;
   QString wynik_pingu;
   int kod_powrotu_pingu;
   QToolButton *szczegoly_pingu;
   QStatusBar *pasek_statusu;
   QToolBar *pasek_narzedzi;

   // watek
   InfoThread *watek;
 public:
   Ui::PaintWindow interfejs;
   PaintWindow(QWidget * parent = 0, Qt::WFlags f = 0 );
   ~PaintWindow(){};
   void paintEvent(QPaintEvent *);
 public slots:
   // pobieranie http
   void zakonczonoPobieraniePliku();

   // komunikacja sieciowa
   void showNetRun();
```

```
    // ping
    void doPing();
    void showPing();
    void odczytajWynikPingu();

    // wątek
    void printNewValue(QString, double);
    void terminateInfoThread();
    void startInfoThread();
};

#endif
```

NET_WINDOW.H

```
#include <QDialog>
#include <QTcpSocket>
#include <QTcpServer>

#include "ui_net_window.h"

class NetWindow: public QDialog {
Q_OBJECT
public:
    Ui::NetWindow interfejs;
    NetWindow(QWidget * parent, Qt::WFlags f);
    ~NetWindow();
 public slots:
    void connectSend();
    void odczytDanych();
    void rozlaczono();
    void odebranoNowePolaczenie();
};
```

INFO_THREAD.H

```
#include <QThread>

class InfoThread: public QThread {
Q_OBJECT
public:
   InfoThread(){};
   ~InfoThread(){};
   void run();
 private:
   QString title, info;
   QWidget* parent;
 signals:
   void putNewValue(QString, double);
};
```

PAINT_WINDOW.CPP

```
#include "paint_window.h"
#include "net_window.h"
```

```
#include "singleton.h"

#define _OBRAZEK_ "./tmp.gif"

// uwaga: to musi być zmienna plikowa a nie klasowa ;-)
QGraphicsEllipseItem *kolko;

void PaintWindow::zakonczonoPobieraniePliku() {
 printf("Pobieranie zakończone %d\n", http->state());
 if (trawa_pobieranie) {
  http->close();
  plik_obrazka->close();

  sleep(2);
  trawa_pobieranie = false;
  info->reject();
 }
}

void PaintWindow::terminateInfoThread() {
 // kończymy wątek
 watek->terminate();
 // ukrywamy pole tekstowe, podobnie można uczynić z dowolnym elementem interfejsu
 interfejs.lineEdit->setVisible(false);
 // mechanizm ten można spróbować wykorzystać do budowy okna z zmiennym interfejsem
 // (np. najpierw formularz logowania a potem dopiero normalny program)
 // w tym celu warto by go użyć do jakiegoś groupBox
 //
 // efekt taki można także próbować osiągnąć z wykorzystaniem setLayout lub ⤸
 addWidget/removeWidget
 // albo przy pomocy QStackedWidget, który potrafi wybierać do wyświetlania jeden z swoich ⤸
 widgetów
}
void PaintWindow::startInfoThread() {
 // pokazujemy pole tekstowe
 interfejs.lineEdit->setVisible(true);
 // startujemy wątek
 watek->start();
}
void PaintWindow::printNewValue(QString info, double val) {
 interfejs.lineEdit->setText(QString::number(val, 'g', 3) + " (" + info + ")");
}

void PaintWindow::showNetRun() {
 NetWindow okno_sieci(0, 0);
 okno_sieci.show();
 okno_sieci.exec();
}

void PaintWindow::showPing() {
 QMessageBox::information (this, "szczegóły pingu", wynik_pingu);
}
void PaintWindow::doPing() {
 pingProces = new QProcess;
 connect( pingProces, SIGNAL(readyReadStandardOutput()), this, SLOT(odczytajWynikPingu()) );
 pingProces->start("/bin/ping", QStringList() << "-c 1" << Singleton::use().adres_ip, ⤸
 QIODevice::ReadOnly);
}
void PaintWindow::odczytajWynikPingu() {
 kod_powrotu_pingu = pingProces->exitStatus();
```

```
 wynik_pingu = pingProces->readAll();
 pingProces->close();

printf ("PING: %d\n\n------------\n%s------------\n\n", kod_powrotu_pingu, wynik_pingu.
 toUtf8().constData() );
 if (kod_powrotu_pingu == 0) {
  pasek_statusu->showMessage("Host osiągalny");
 } else {
  pasek_statusu->showMessage("Host nie osiągalny");
 }
}

PaintWindow::PaintWindow(QWidget * parent, Qt::WFlags f):QMainWindow(parent, f) {
 // skonfigurowanie interfeju w oparciu o obiekt klasy wygenerowanej na podstawie pliku .ui
 interfejs.setupUi(this);

/// OBRAZEK przez HTTP
 // sprawdzamy czy jest obrazek gdy nie ma pobieramy przez http
 plik_obrazka = new QFile(_OBRAZEK_);
 if (! plik_obrazka->exists()) {
  // otwieramy plik do zapisu
  plik_obrazka->open(QIODevice::WriteOnly);

  // tworzymy QHttp i powiązujemy sygnał done
  http = new QHttp(this);
  connect(http, SIGNAL( done(bool) ), this, SLOT( zakonczonoPobieraniePliku() ));

  // ustawiamy host
  http->setHost("www.google.com");
  // wysyłamy żadanie http GET,
  // bardziej zaawansowane operacje na nagłówkach umożliwia nam QHttpRequestHeader
  http->get("/images/logo.gif", plik_obrazka);

  trawa_pobieranie = true;
  info = new QMessageBox("Please wait ...", "Proszę czekać na pobranie pliku", QMessageBox::
  Information,\
   QMessageBox::Ok, QMessageBox::NoButton, QMessageBox::NoButton);
  info->setModal(false);
  while(trawa_pobieranie)
   info->exec();
 }

/// RYSOWANIE po okienku
 /// wiele przykładów jak co rysować w http://doc.trolltech.com/qpainter.html

// załadowanie obrazka
 obrazek = new QImage(_OBRAZEK_);

// pędzelek i rysik
 pedzel = new QBrush(QColor(255,83,24,125));
 rysik = new QPen(*pedzel, 16);

// robimy rysunek
 rysunek = new QPicture();
 QPainter pole_rysunku;
 // przywiązanie pola po którym rysujemy do naszego rysunku
 pole_rysunku.begin(rysunek);
 // ustawiamy rysik i rysujemy elipsę
 pole_rysunku.setPen(*rysik);
 pole_rysunku.drawEllipse(10,20, 180,70);
```

```
 // zmieniamy rysik na predefiniowany i rysujemy prostą
 pole_rysunku.setPen(Qt::green);
 pole_rysunku.drawLine(QPoint(0,0), QPoint(40,30));
 // zmieniamy rysik na oparty na predefiniownym pędzlu i zadanej grubości
 // i rysujemy kolejną linię
 // warto zauważyć że po wstawieniu tego obrazka wszystkie pozycje są wyrażone
 // względem punktu wstawienia, ale rysunek nie ma ograniczenia rozmiarowego
 // (jest tak duży jak narysujemy)
 pole_rysunku.setPen(QPen(Qt::red, 5));
 pole_rysunku.drawLine(QPoint(40,30), QPoint(60,-30));
 // kończymy przygotowywanie naszego rysunku
 pole_rysunku.end();

// zmieniamy pedzelek i rysik
 pedzel->setTextureImage(*obrazek);
 // jak dalej zobaczymy możemy pisać przy pomocy obrazka 9tekstury0 ...
 rysik->setBrush(*pedzel);

/// RYSOWANIE w QGraphicsView
 /// wiele przykładów i dokładny opis w http://doc.trolltech.com/graphicsview.html

QGraphicsScene *scenka = new QGraphicsScene();
 scenka->addRect(QRectF(0, 0, 100, 100));
 kolko = scenka->addEllipse(QRectF(0, 0, 100, 100), QPen(Qt::red, 5));

interfejs.graphicsView->setScene(scenka);
 interfejs.graphicsView->show();

/// można też programować w qt WIELOWATKOWO

// wątek informacji i rzeczy z nim związne
 watek = new InfoThread();
 connect(watek, SIGNAL( putNewValue(QString, double) ), this, SLOT( printNewValue(QString, ↩
 double) ));
 // gdyby nie to że chcemy ukryć pole tekstowe można by od razu wywolać sloty terminate ↩
 oraz start z watek
 connect(interfejs.radio_stop, SIGNAL( clicked(bool) ), this, SLOT( terminateInfoThread() ));
 connect(interfejs.radio_start, SIGNAL( clicked(bool) ), this, SLOT( startInfoThread() ));
 // zaznaczamy przycisk radio - umieszczenie zespołu tych przycisków w ramce
 // gwarantuje że zaznaczony może być conajwyżej jeden
 interfejs.radio_start->setChecked(true);
 watek->start();

/// MENU TRYBU SIECIOWEGO
 connect(interfejs.actionRun, SIGNAL( triggered() ), this, SLOT( showNetRun() ));

/// PASEK NARZEDZI i PASEK STATUSU dla PINGu
 Singleton::use().adres_ip = "127.0.0.1";
 Singleton::use().port = 8080;

pasek_narzedzi = addToolBar ("Jedyny pasek narzędzi");
 QAction *action_ping = new QAction(/* tu jako pierwszy argument można podac ikonę, */ ↩
"Ping", this);
 // tak zdefiniowaną akcję możemy też podpinać pod pozycje menu głównego, menu ↩
 kontekstowego czy przycisków
 // jest to wygodne przy większych programach gdzie pozwalamy na konigurowanie menu i ↩
 toolbarów
 action_ping->setStatusTip ("Wykonuje ping");
 pasek_narzedzi->addAction( action_ping );
 connect(action_ping, SIGNAL( triggered() ), this, SLOT( doPing() ));
```

```
pasek_statusu = statusBar();
 szczegoly_pingu = new QToolButton();
 szczegoly_pingu->setText("pokaż wynik pingu");
 QFont czcionka;
 czcionka.setPointSize(6);
 szczegoly_pingu->setFont(czcionka);
 connect(szczegoly_pingu, SIGNAL( clicked(bool) ), this, SLOT( showPing() ));
 pasek_statusu->addPermanentWidget(szczegoly_pingu);
 pasek_statusu->showMessage("Nie wykonano pingu");
}

// reinterpereyujemy funkcję mousePressEvent klasy QGraphicsView
// wykorzystywanej do wyświetlania naszego "zaawansowanego" rysunku
// dzieki temu jesteśmy w stanie przechwycić zdarzenia związane kliknięciem na nim myszą
void QGraphicsView::mousePressEvent (QMouseEvent *event) {
 // pozycja trafienia
 QPoint pozycja = event->pos();
 QPointF pozycja2 = this->mapToScene(pozycja);
 printf( "Klikneliśmy w (%d, %d) czyli na obrazek na graphicsView w (%f, %f)\n",\
  pozycja.x(), pozycja.y(), pozycja2.x(), pozycja2.y() );

// w co trafilismy
 QList<QGraphicsItem *> trafione = items(event->pos());
  // uwaga używamy pozycji względem okna a nie sceny !!!
 for (int i = 0; i<trafione.size(); i++) {
  if (trafione[i] == kolko)
   puts("trafiliśmy w kółko");
 }
}

// funkcja przerysowująca okienko gdy zachodzi taka potrzeba ...
// w niej należy umieszczać rysowanie po okienku
// ze względów wydajnościowych wartio korzystać z rysowania przygotowanych wcześniej ⤸
rysunków
void PaintWindow::paintEvent(QPaintEvent *) {
 QPainter painter(this);

//wyświetlamy napis
 painter.setPen(*rysik);
 painter.setFont(QFont("Times", 44));
 painter.drawText(QPoint( 280, 80 ), "Ala ma kota");

// rysujemy wczytany obrazek
 painter.drawImage(QPoint( 320, 120 ), *obrazek);

// rysujemy przygotowany wcześniej rysunek
 painter.drawPicture(QPoint( 280, 160 ), *rysunek);
}
```

NET_WINDOW.CPP

```
#include "net_window.h"
#include "singleton.h"


QTcpSocket *gniazdko_tcp;
QTcpServer *serwerek_tcp;
```

```
// udostępnione jest też udp - używamy dość podobnie jak QTcpSocket
// więcej szczegółów http://doc.trolltech.com/qudpsocket.html

void NetWindow::connectSend() {
 // jeżeli nie połączono to łączymy
 if (gniazdko_tcp == NULL && serwerek_tcp == NULL) {
  // aktualizujemy wartości zapisane w konfiguracji run-time
  Singleton::use().port = interfejs.spin_port->value();
  Singleton::use().adres_ip = interfejs.txt_ip->text();
  // blokujemy interfejs
  interfejs.spin_port->setDisabled(true);
  interfejs.txt_ip->setDisabled(true);
  interfejs.check_listen->setDisabled(true);

  if (interfejs.check_listen->checkState() == Qt::Unchecked) {
   puts("client mode");
   // tworzymy gniazdo
   gniazdko_tcp = new QTcpSocket(); // to nie to samo co QAbstractSocket(QTcpSocket::
   TcpSocket)
   connect(gniazdko_tcp, SIGNAL( readyRead() ), this, SLOT( odczytDanych() ));
   connect(gniazdko_tcp, SIGNAL( disconnected() ), this, SLOT( rozlaczono() ));
   // łączymy się z hostem
   gniazdko_tcp->connectToHost(Singleton::use().adres_ip, Singleton::use().port, QIODevice::
   ReadWrite);
   // czekamy na połączenie - konieczne jeżeli chcemy od razu wysłać dane
   gniazdko_tcp->waitForConnected();
  } else {
   puts("server mode");
   // tworzymy serwer
   serwerek_tcp = new QTcpServer(this);
   serwerek_tcp->setMaxPendingConnections(1);
   connect(serwerek_tcp, SIGNAL(newConnection()), this, SLOT(odebranoNowePolaczenie()));
   // ustawiamy nasłuchiwanie na wszystkich adresach i zadanym porcie
   serwerek_tcp->listen(QHostAddress::Any, Singleton::use().port);
   // kolorowy komunikat
   interfejs.txt_recived->setTextColor(Qt::green);
   interfejs.txt_recived->append("[[Serwer uruchomiony]]\n");
  }
 }
 if (gniazdko_tcp != NULL) {
  // wysyłamy dane z interfejs.txt_send
  gniazdko_tcp->write((interfejs.txt_send->toPlainText() + "\n").toUtf8());
  // wypisujemy w oknie historii
  interfejs.txt_recived->setTextColor(Qt::gray);
  interfejs.txt_recived->append(interfejs.txt_send->toPlainText() + "\n");
  // czyścimy pole tekstowe
  interfejs.txt_send->setText("");
  // ustawiamy na nim focus
  interfejs.txt_send->setFocus(Qt::OtherFocusReason);
 }
}

void NetWindow::odebranoNowePolaczenie() {
 if (gniazdko_tcp == NULL) {
  // przywiązujemy odebrane połączenie do gniazdka
  gniazdko_tcp = serwerek_tcp->nextPendingConnection();
  connect(gniazdko_tcp, SIGNAL( readyRead() ), this, SLOT( odczytDanych() ));
  connect(gniazdko_tcp, SIGNAL( disconnected() ), this, SLOT( rozlaczono() ));
  // kolorowy komunikat
  interfejs.txt_recived->setTextColor(Qt::blue);
```

```
    interfejs.txt_recived->append("[[Zestawiono nowe połączenie]]\n");
  } else {
    // przywiązujemy odebrane połączenie do tymczasowego gniazdka
    QTcpSocket *gniazdko_tcp_tmp = serwerek_tcp->nextPendingConnection();
    connect(gniazdko_tcp_tmp, SIGNAL(disconnected()), gniazdko_tcp_tmp, SLOT(deleteLater()));
    // wysyłamy odpowiedź
    gniazdko_tcp_tmp->write("ten serwer obsługuje tylko jedno połączenie\n");
    // zamykamy połączenie
    gniazdko_tcp_tmp->disconnectFromHost();
    // kolorowy komunikat
    interfejs.txt_recived->setTextColor(Qt::red);
    interfejs.txt_recived->append("[[Odrzucono nowe połączenie]]\n");
  }
}

void NetWindow::odczytDanych() {
  // wypisujemy otrzymane dane w kolorze czarnym
  interfejs.txt_recived->setTextColor(Qt::black);
  interfejs.txt_recived->append(gniazdko_tcp->readAll());
}

void NetWindow::rozlaczono() {
  // czyścimy gniazdko, tak dla pewności działania warunków startu
  gniazdko_tcp = NULL;
  // jeżeli nie działa serwer odblokowujemy interfejs
  if (serwerek_tcp == NULL) {
    interfejs.spin_port->setDisabled(false);
    interfejs.txt_ip->setDisabled(false);
    interfejs.check_listen->setDisabled(false);
  }
  // kolorowy komunikat
  interfejs.txt_recived->setTextColor(Qt::red);
  interfejs.txt_recived->append("[[Rozłączono]]\n");
}

NetWindow::NetWindow(QWidget * parent = 0, Qt::WFlags f = 0 ):QDialog(parent, f) {
  interfejs.setupUi(this);

// wyświetlamy wartości zapisane w konfiguracji run-time
  interfejs.txt_ip->setText(Singleton::use().adres_ip);
  interfejs.spin_port->setValue(Singleton::use().port);

// wynullowanie zmiennych i przypisanie akcji do guzika
  gniazdko_tcp = NULL; serwerek_tcp = NULL;
  connect(interfejs.btn_send, SIGNAL( clicked(bool) ), this, SLOT( connectSend() ));
}

NetWindow::~NetWindow() {
  // jeżeli aktywne to zamykamy gniazdko
  if (gniazdko_tcp != NULL) {
    gniazdko_tcp->disconnectFromHost();
  }
  // jeżeli działa serwer to go kończymy
  if (serwerek_tcp != NULL) {
    serwerek_tcp->close();
  }
}
```

INFO_THREAD.CPP

```
#include "info_thread.h"

void InfoThread::run() {
 // UWAGA: wątki (poza głównym wątkiem GUI) nie mogą produkować okien itp
 double rnd_val = 0;
 while (1) {
  // obliczamy nową wartość
  rnd_val += 0.5;
  // wysyłamy sygnał - obliczona nowa wartość
  printf("InfoThread: %f\n", rnd_val);
  emit putNewValue("Test Random Value", rnd_val);
  // odpoczywamy sobie 10 sekund
  QThread::sleep(10);
 }
}
```

GRAPHICS_NET_MENU_TOOLBAR.CPP

```
/**
 * Program prezentujący podstawy programowania w Qt4 - prezentuje sposoby rysowania,
 * tworzenie menu i pasków narzędzi, wywoływanie i komunikowanie się z zewnętrznymi ⤸
 aplikacjami,
 * korzystanie z http oraz gniazd sieciowych, zmienne aplikacyjne (singleton), watki
 *
 * jest plik główny programu z funkcją main
 * poostałe elementy umieszczone są w plikach klas obsługujących poszczególne okienka ⤸
 oraz
 * singleton i wątek
 *
 * KOMPILACJA: qmake -project && echo 'QT += network' >> *.pro && qmake && make
 * POSPRZĄTANIE: make distclean && rm -f *.pro tmp.gif
 **/

// nagłówki Qt
#include <QApplication>
#include <QTextCodec>

// nagłowek klasy związanej z głównym okienkiem aplikacji
#include "paint_window.h"

int main(int argc, char *argv[]) {
 QApplication aplikacja(argc, argv);

// ustawiamy kodowanie dla wyświetlanych i dla przetwarzanych napisów
 QTextCodec::setCodecForTr (QTextCodec::codecForName ("UTF-8"));
 QTextCodec::setCodecForCStrings (QTextCodec::codecForName ("UTF-8"));

// tworzymy okienko w oparciu o klase PaintWindow
 PaintWindow okienko(0, 0);

// wyświelamy okienko
 okienko.show();

// uruchamiamy aplikację
 return aplikacja.exec();
}
```

MyWindow.cpp

```
#include <Application.h>

#include "MyWindow.h"

// zmienna kontrolna z ktorej pozniej skorzystamy
bool wybrano_drugie_menu=0;

/***** Constructor ****/
MyWindow::MyWindow(BRect frame,const char *title,window_look look,window_feel feel,uint32 ⤸
flags,uint32 workspace)
: BWindow(frame,title,look,feel,flags,workspace) {
  // funkcja: BRect(gorny_lewy_rog_X, gorny_lewy_rog_Y, dolny_prawy_rog_X, dolny_prawy_rog_Y)
  // - pozycjonuje i nadaje rozmiar obiektowi

// dodaje punkty menu rozwijanego
  // na poczatku pojawi sie tekst "wpisywana nazwa menu"
  select_menu = new BMenu("wpisywana nazwa menu");
  select_menu->AddItem(new BMenuItem("opcja 1", new BMessage(MSG_SELECT_1)));
  select_menu->AddItem(new BMenuItem("opcja 2", new BMessage(MSG_SELECT_2)));

// to "zaklecie" pozwoli nam na wybieranie opcji w menu
  select_menu->SetLabelFromMarked(true);
  // mozemy tez wybrac opcje ktora bedzie wybrana na poczatku
  // jednak wtedy bez znaczenia jest polecenie: BMenu("wpisywana nazwa menu")
  // gdyz na starcie bedzie widoczna ta ustalona ponizej wartosc domyslna
  //(select_menu->ItemAt(0))->SetMarked(true);

// tworzymy i umieszczamy menu rozwijane
  select = new BMenuField(BRect(15, 45, 300, 55), "Menu", "Wyswietlana etykieta menu:", ⤸
  select_menu);
  AddChild(select);


// dodaje przysisk - przekazuje on komunikat zdefiniowany w MSG_PRZYSISK_1
  AddChild( new BButton( BRect(15, 10, 100, 40), "Przycisk", "Nazwa przycisku",
   new BMessage( MSG_PRZYSISK_1 ) ) );
}

// okreslam reakcje na przekazywane komunikaty
void MyWindow::MessageReceived(BMessage *message)
{
  int wynik;

switch (message->what){
  // gdy zostal odebrany komunikat zdefiniowany w MSG_PRZYSISK_1
  case MSG_PRZYSISK_1:

    // wyswietl komunikat BAlert:
    // ponizej jest B_IDEA_ALERT z trzema przyciskami - ulozone w rownych odstepach: ⤸
    B_EVEN_SPACING
    alert_1 = new BAlert("", "wybierz rodzaj komunikatu:", "alert", "warning", "info",
     B_WIDTH_AS_USUAL, B_EVEN_SPACING, B_IDEA_ALERT);
    wynik = alert_1->Go();
    // mozna by to zapisac krocej: wynik = (new BAlert("", "wybierz rodzaj komunikatu:",
     "alert", "warning", "info", B_WIDTH_AS_USUAL, B_OFFSET_SPACING, B_IDEA_ALERT) )->Go();

    // moga byc tez B_STOP_ALERT z dwoma przysiskami
    if (wynik==0)
```

```
    (new BAlert("", "treść komunikatu", "OK 1", "OK 2", NULL, B_WIDTH_AS_USUAL,
     B_OFFSET_SPACING, B_STOP_ALERT))->Go();
  // albo B_WARNING_ALERT z trzema
  if (wynik==1)
    (new BAlert("", "treść komunikatu", "OK 1", "OK 2", "OK 3", B_WIDTH_AS_USUAL,
     B_OFFSET_SPACING, B_WARNING_ALERT))->Go();
  // i najprostrzy - - standardowy (B_INFO_ALERT)
  if (wynik==2)
    (new BAlert("", "treść komunikatu", "OK"))->Go();

  // BAlert z dwoma guzikami moze wygladac tez tak (wyswietli sie gdy wybierzemy druga opcje ⤸
  menu):
  if (wybrano_drugie_menu) (new BAlert("", "wybrałeś opcję nr 2", "OK 1", "OK 2", NULL,
   B_WIDTH_AS_USUAL, B_EVEN_SPACING, B_STOP_ALERT))->Go();

  // koncze instrukcje case
  break;

// gdy wybrana zostanie pierwsza pozycja menu
 case MSG_SELECT_1:
  wybrano_drugie_menu =0;
  break;

// gdy wybrana zostanie pierwsza pozycja menu
 case MSG_SELECT_2:
  wybrano_drugie_menu =1;
  break;

default:
  BWindow::MessageReceived(message);
  break;
 }
}

/***** QuitRequested ****/
bool MyWindow::QuitRequested()
{
 // przekazuje sygnal nakazujacy zakonczenie - takie rozwiazanie wymaga #include <Application.⤸
 h>
 be_app->PostMessage(B_QUIT_REQUESTED);
 return(true);
}
```

MyWindow.h

```
// kontrola jednorazowego wczytywania
#ifndef _MYWINDOW_H
#define _MYWINDOW_H

/* teraz to co naprawde istotne ... */

// wczytujemy nagłówki dla elementow beos'owego GUI
 #include <InterfaceKit.h>

#define MSG_PRZYSISK_1 'CONV'
 #define MSG_SELECT_1 'SS1'
 #define MSG_SELECT_2 'SS2'
```

```
// deklarujemy to z czego bedziemy korzystac w wlasciwym pliku - MyWindow.cpp
 class MyWindow : public BWindow
{
  public:   // publics functions
   MyWindow(BRect frame,const char *title,window_look look,
    window_feel feel,uint32 flags,uint32 workspace = B_CURRENT_WORKSPACE);
   void MessageReceived(BMessage *message);
   virtual bool  QuitRequested();

  protected: // protected members
   BAlert *alert_1;
   BMenu *select_menu;
   BMenuField *select;
 };

#endif
```

MAIN.CPP

```
#include "main.h"
#include "MyWindow.h"

int main(void)
{
 moja_aplikacja application("application/moja_aplikacja");
 application.Run();

return 0;
}

moja_aplikacja::moja_aplikacja(const char *signature) : BApplication(signature)
{
 // lokuje i wyświetlam okno programu
 _mywindow = new MyWindow(BRect(232,148,569,294),"Tytuł okna",
  B_DOCUMENT_WINDOW_LOOK, B_NORMAL_WINDOW_FEEL,B_NOT_ZOOMABLE,0);
 _mywindow->Show();
}
```

MAIN.H

```
#ifndef _MOJA_APLIKACJA_H
#define _MOJA_APLIKACJA_H

#include <Application.h>

class MyWindow;

class moja_aplikacja : public BApplication
{
  public:   // publics functions
   moja_aplikacja(const char *signature);

  protected: // protected members
   MyWindow  *_mywindow;
 };
```

```
#endif
```

NCURSES_1.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody związane z sterowaniem wyświetlaniem na terminalu tekstowym
 * plik ten wykorzystuje meodę opartą w całości na bibliotece termcap
 * (przy tej metodzie mamy pewność że zawsze powinna zadziałać)
 *
 */

#include <termcap.h>

#include <stdio.h>

#include <string.h> /* for memset() */
#include <stdlib.h> /* for getenv() */

/* Wymagane -ltermcap lub -lncurses przy linkowaniu */

static int max_x, max_y;
static char *tg_cm;

/* inicjalizacja terminala */
void inicjalizacja_terminalu() {
/* oczywiscie wiekszosc tych funkcji moze nie zakonczyc sie powodzeniem co nalezaloby ⤸
sprawdzac ... */
/* pobranie infa o terminalu */
tgetent (NULL, getenv ("TERM"));

/* zczytanie parametrow terminala */
max_x=tgetnum("co");
 max_y=tgetnum("li");

tg_cm=tgetstr("cm", 0); /* pozycja kursora */
}

/* wyczyszczenie terminala */
void wyczysc_terminal(void) {

tputs(tgetstr("cl",0), 1, putchar);
}

/* wypisanie w zadanym punkcie ekranu,
   warto zwrócić uwagę że jest to sposób tworzenia funkcji opartych na instrukcjach ⤸
   przeprocesora
   - przed kompilacją kod funkcji zostanie wstawiony w mijsca jej wywołań */
#define put_s(x,y,s) tputs ( tgoto(tg_cm,y,x) , 1, putchar); printf(s);

/* funkcja glowna */
int main(void) {
 inicjalizacja_terminalu();

wyczysc_terminal();

put_s(2,8,"@")
```

```
/* oczekiwanie na nacisniecie q */
while (getchar()!='q');

return 0;
}
```

NCURSES_2.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody związane z sterowaniem wyświetlaniem na terminalu tekstowym
 * plik ten wykorzystuje meodę opartą w całości na bibliotece terminfo
 * (z tej biblioteki korzysta też ncurses, czyli jest to taka bardziej niskopoziomowa metoda)
 *
 */

#include <termios.h>
#include <term.h>

#include <stdio.h>

#include <fcntl.h>
#include <unistd.h>
#include <string.h> /* for memset() */

/* Wymagane -lncurses przy linkowaniu */

struct termios old_termios, new_termios;
static int max_x, max_y;
static char *tig_cup;

/* inicjalizacja terminala */
void inicjalizacja_terminalu() {
/* oczywiscie wiekszosc tych funkcji moze nie zakonczyc sie powodzeniem co nalezaloby ⤸
sprawdzac ... */
/* otwarcie terminala */
int t;
  /* otwieramy plik-urządzenie terminala i zastępujemy nim standardowe wyjście */
t=open("/dev/tty",O_RDWR,0); dup2(t,1); close(t);

setupterm(NULL,1,&t);

/* zapis starych ustawien terminala */
tcgetattr(1,&old_termios);

/* wyczyszczenie ustawien terminala - nie bedzie widac wpisywanych znakow ... */
memset(&new_termios,0,sizeof(new_termios));
 tcsetattr(1,TCSANOW,&new_termios);

/* zczytanie parametrow terminala */
max_x=tigetnum("cols");
 max_y=tigetnum("lines");

tig_cup=tigetstr("cup"); /* pozycja kursora (patrz man 5 terminfo),
   inne to m.in. "bold" - przełącznik w tryb gruby, "setf" - ustawianie koloru tła ... */
}

/* wyczyszczenie terminala */
```

```
void wyczysc_terminal(void) {
 putp(tparm(tig_cup,0,0)); /* idz na sam poczatek */
putp(tparm(tigetstr("ed")));
}

/* wypisanie w zadanym punkcie ekranu,
     warto zwrócić uwagę że jest to sposób tworzenia funkcji opartych na instrukcjach ↩
     przeprocesora
     - przed kompilacją kod funkcji zostanie wstawiony w mijsca jej wywołań */
#define put_s(y,x,s) putp(tparm(tig_cup,y,x)); printf(s);

/* funkcja glowna */
int main(void) {
 inicjalizacja_terminalu();

wyczysc_terminal();

put_s(2,8,"@")

/* oczekiwanie na nacisniecie q */
while (getchar()!='q');

/* przywrocenie ustawien terminala */
tcsetattr(1,TCSAFLUSH,&old_termios);

return 0;
}
```

NCURSES_3.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody związane z sterowaniem wyświetlaniem na terminalu tekstowym
 * plik ten wykorzystuje meodę opartą w całości na bibliotece ncurses
 *
 */

#include <ncurses.h>
/* Wymagane -lncurses przy linkowaniu */

int main(void) {
 initscr(); /* inicjalizacja */
noecho(); /* nie bedzie widac wpisywanych znakow */
curs_set(false); /* nie pokazuj kursora */

/* można by jeszcz używać m.in.:

 cbreak(); - wylacza buferowanie linii
 nonl(); - sterowanie wejsciem
 intrflush(stdscr, FALSE); - sterowanie wejsciem
 keypad(stdscr, TRUE); - sterowanie wejsciem

 */

/* wypisz znak na zadanej pozycji */
mvaddch(2,8,'@');

/* oczekiwanie na nacisniecie q */
```

```
while (getch()!='q');

curs_set(true); /* pokazuj normalnie kursor */
endwin(); /* zakoncz */

return 0;
}
```

Multimedia

Dźwięk w Linuxie obsługiwany jest najczęściej przez sterowniki Alsa lub OSS - w obu przypadkach dostępne jest urządzenie `/dev/dsp` jednak w przypadku Alsa nie jest ono natywnym sposobem obsługi tego sterownika. Często także powyżej sterownika funkcjonują serwery dźwięku takie jak "jack" lub "arts"

Z kolei **SDL** jest biblioteka multimedialna, obsługująca grafikę 2D i 3D, dźwięk, urządzenia wejścia/wyjścia. W zastosowaniach graficznych wykorzystuje **OpenGL** (niskopoziomowy interfejs graficzny). OpenGL posiada również własne wyżej poziomowe biblioteki (GLU, GLUT, GLUI, ...). Można powiedzieć że SDL stanowi swego rodzaju wieloplatformowy odpowiednik dla DirectX (OpenGL jest rozwiązaniem konkurencyjnym dla Direct3D).

W tym miejscu warto też wspomnieć że istnieją również wyżej poziomowe, wieloplatformowe biblioteki wyspecjalizowane w tworzeniu gier, takie jak: **Allegro** (grafika 2D), **Crystal Space** (grafika 3D), ...

- dsp.c - program kopiujący dane z /dev/dsp do serwera artsd (kiedys kopiowal do /dev/dsp); demonstruje korzystanie z tego pliku oraz rozwiązuje problem oglądania TV z tunera z wykorzystaniem (zintegrowanych) kart dźwiękowych opartych na CMI9880
- alsa_copy.c - program o funkcjonalności dsp.c tyleże korzystający wyłącznie z biblioteki asound (ALSA) a zatem umożliwiający korzystanie z dmix
- openal.c - program prezentuje podstawy korzystania z biblioteki OpenAL służącej do implementacji dźwięku (przestrzennego) w multimediach
- pokaz_slajdow_sdl.c - program demonstrujący korzystanie z SDL (wyświetlanie bitmap, obsługa zdarzeń - klawiatura i tworzenie własnych, timery, czyszczenie ekranu)
- rysownik_sdl.c - program demonstrujący korzystanie z SDL (rysowanie pikseli, obsługa zdarzeń - klawiatura i mysz, zapis obrazu do bitmapy)
- read_png.c - program demonstrujący korzystanie z libPNG do wczytywania obrazkow
- obraz_kontrolny.cpp - program generujący z wykorzystaniem SDL obraz kontrolny (umieszcza stosowne napisy na podanym pliku graficznym), domonstruje również użycie LibVLC jako wyjścia obrazu (zobacz w sieci: przykład użycia libvlc jako wejścia obrazu[1007]), zamieszczam też archiwum[1008] z obrazami kontrolnymi do wykorzystania w tym programie.
- zamieszczam zestaw przykładowych programów obrazujących korzystanie z OpenGL:
  - kod odpowiedzialny za inicjalizację okana OpenGL - pokazujący różnice w korzystaniu z glut, sdl i glx opengl-main-glut_sdl_glx.c lub dający więcej funkcjonalności ale tylko z wykorzystaniem glut opengl-main-glut.c
  - kod odpowiedzialny za inicjalizację OpenGL - opengl-init.c
  - kod odpowiedzialny za rysowanie sceny OpenGL - opengl-draw-simple.c (podstawy) lub opengl-draw-vertex.c (obsługa bardziej zaawansowanych metod rysowanie) lub opengl-draw-light.c (obsługa świateł)
  - kod odpowiedzialny za ładowanie shaderów z wykorzystaniem mmap opengl-load_shader.c

  Zobacz także mój projekt podstawa silnika 3d.

Zobacz w Sieci: Strona domowa Janusza Ganczarskiego - OpenGL[1009], OpenGL Shading Language Tutorials and Demos[1010], GLSL Tutorial[1011], Grafika 3D - OpenGL[1012], Tutoriale do OpenGL na spacesimulator.net[1013], Kurs programowania GLUT[1014], Jaka metoda generowania dynamicznych cieni?[1015], oficjalna dokumentacja OpenGL[1016] (man pages, specifications, ... for OpenGL, GLUT, GLU, GLX, ...), Shader[1017], Opengl[1018], GLUT[1019], OpenGL Shading Language[1020], OpenGL Shading Language @ Wikibooks[1021], OpenCL[1022]. Interesujące jest także wykorzystanie dostępnych na wolnych licencjach bibliotek / silników obsługi grafiki trójwymiarowej, takich jak OGRE[1023], Irrlicht[1024], Crystal Space[1025]. Zobacz w Sieci: Kurs Irrlicht, C++[1026], OGRE @ Wikibooks[1027].

DSP.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje korzystanie z pliku /dev/dsp oraz serwera dźwiękowego arts,
 *
 * kompilacja: gcc ‘artsc-config –cflags‘ ‘artsc-config –libs‘ dsp1.c
 *
 * praktycznym wykorzystaniem tego programu jest implementacja pętli pomiędzy
 * wejściem liniowym a wyjściem głośnikowym w karcie dźwiękowej opartej o ICH6 + ⤦
 CMI9880
 *
 * pierwotna wersja tego programu była tylko prostą modyfikacja programu obrazującego
 * korzystanie z /dev/dsp - parrot.c ( ftp://adlib.rsch.tuis.ac.⤦
 jp/pub/Linux/multi_media_guide/ )
 * obecnie kod programu został całkowicie przepisany oraz wzbogacony o obsługę serwera ⤦
 dźwięku arts
 *
 * więcej o arts: http://www.arts-project.org/doc/mcop-doc/artsc.html
 *
 */

#include <linux/soundcard.h>
#include <kde/artsc/artsc.h>

#include <stdlib.h>
#include <fcntl.h>
#include <stdio.h>
#include <errno.h>

#define SAMPLE_PER_SECOND 48000
#define SAMPLE_SIZE 16
#define CHANNELS 2

int tmp;

int main() {

/// inicjalizacja arts
 arts_init();
 arts_stream_t arts_stream = arts_play_stream( SAMPLE_PER_SECOND, SAMPLE_SIZE, CHANNELS, ⤦
 "arts_dsp_copy" );
 arts_stream_set( arts_stream, ARTS_P_BUFFER_TIME, 50 );
     // bufor na 50 ms gdyby nie bylo tej instrukcji to byloby duze opoznienie
 arts_stream_set( arts_stream, ARTS_P_BLOCKING, 0 );
 int packet_size = arts_stream_get( arts_stream, ARTS_P_PACKET_SIZE );

/// bufor
 char * buf;
 buf = malloc(packet_size);

/// otwarcie /dev/dsp do odczytu
 int source_d = open("/dev/dsp", O_RDONLY);
 if (source_d < 0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }

/// konfiguracja /dev/dsp
 // funkcja ioctl w przypadku bledu zwraca < 0 i ustawia errno
```

```
 // ale dla uproszczenia nie obsługuję tu tych błedów
 tmp = SAMPLE_PER_SECOND;
 ioctl(source_d, SOUND_PCM_WRITE_RATE, &tmp);
 tmp = SAMPLE_SIZE;
 ioctl(source_d, SOUND_PCM_WRITE_BITS, &tmp);
 tmp = CHANNELS;
 ioctl(source_d, SOUND_PCM_WRITE_CHANNELS, &tmp);

/// pętla kopiująca
 while (1) {
  tmp = read(source_d, buf, packet_size);
  arts_write(arts_stream, buf, tmp);
  // kiedys program ten zapisywal tutaj do /dev/dsp zamiast do arts ...
 }

/// zakanczanie programu
 // w zasadzie to to powinno byc w obsludze sygnalu SIGINT (Ctrl+C) i SIGTERM (program kill ⤸
 bez opcji)
 // ale i tak przy zakonczeniu to zostanie zwolnione ...
 close(source_d);
 arts_close_stream( arts_stream );
 arts_free();
 free(buf);
}
```

ALSA_COPY.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje korzystanie z blibloteki dźwiękowej ALSA (odczyt i zapis dźwięku do ⤸
 użądzenia)
 *
 * kompilacja: gcc -lasound alsa_copy.c
 *
 * praktycznym wykorzystaniem tego programu jest implementacja pętli pomiędzy
 * wejściem liniowym a wyjściem głośnikowym w karcie dźwiękowej opartej o ICH6 + ⤸
 CMI9880
 *
 * jest to program który może zastąpić dsp.c gdy nie chcemy używać arts do miksowania ⤸
 dźwięku
 * a też nie chcemy zajmować /dev/dsp (co ma miejsce gdy tamten program pisze zamiast
 * do arts do tego użądzenia) gdyż mikser programowy systemu ALSA (dmix) nie obsługuje
 * miksowania z /dev/dsp tworzonego przez snd_pcm_oss (aby korzystać z programów ⤸
 wymagających
 * OSS poprzez dmix nalezy uzywać ich z aoss oraz umieścić stosowny wpis w /etc/asound.⤸
 conf
 * dotyczący miksowania użądzenia dsp, alternatywną droga jest korzystanie z serwera ⤸
 dźwięku JACK)
 *
 * o dmix'ie - http://alsa.opensrc.org/DmixPlugin
 * o API asoundlib - http://www.alsa-project.org/alsa-doc/alsa-lib/
 *
 */

#include <alsa/asoundlib.h>

static char *alsa_device = "default";
snd_pcm_t *h_in;
```

```
snd_pcm_t *h_out;
unsigned short buf[16*512];

int err;

int main(void) {
 // otwarcie urzadzenia
 err = snd_pcm_open(&h_in, alsa_device, SND_PCM_STREAM_CAPTURE, 0);
 if (err < 0) {
  printf("ERROR (1a): %s\n", snd_strerror(err));
  exit(EXIT_FAILURE);
 }

err = snd_pcm_open(&h_out, alsa_device, SND_PCM_STREAM_PLAYBACK, 0);
 if (err < 0) {
  printf("ERROR (1b): %s\n", snd_strerror(err));
  exit(EXIT_FAILURE);
 }


// ustawienie parametrow urzadzenia
 err = snd_pcm_set_params(h_in, SND_PCM_FORMAT_U16, SND_PCM_ACCESS_RW_INTERLEAVED, 1, 16000, ↩
 1, 800000);
 if (err < 0) {
  printf("ERROR (2a): %s\n", snd_strerror(err));
  exit(EXIT_FAILURE);
 }

err = snd_pcm_set_params(h_out, SND_PCM_FORMAT_U16, SND_PCM_ACCESS_RW_INTERLEAVED, 1, 16000, ↩
 1, 800000);
 if (err < 0) {
  printf("ERROR (2b): %s\n", snd_strerror(err));
  exit(EXIT_FAILURE);
 }

// pętla kopiująca
 while(1) {
  err = snd_pcm_readi (h_in, buf, 8*512);
  if (err == -EPIPE) {
   puts("ERROR (3a): snd_pcm_prepare\n");
   // zapobiega wywalaniu się programu na błędach odczytu z urządzenia
   snd_pcm_prepare(h_in);
  } else if (err < 0) {
   printf("ERROR (3a): %s\n", snd_strerror(err));
  }

  err = snd_pcm_writei(h_out, buf, err);
  if (err < 0) {
   err = snd_pcm_recover(h_out, err, 0);
  }
  if (err < 0) {
   printf("ERROR (3b): %s\n", snd_strerror(err));
   break;
  }
 }

snd_pcm_close(h_in);
 snd_pcm_close(h_out);
}
```

OPENAL.C

```c
/**
 * program prezentuje podstawy korzystania z biblioteki OpenAL
**/

// nagłówki OpenAL
#include <AL/al.h>
#include <AL/alc.h>
#include <AL/alut.h>

#include <stdlib.h>

// dokumentacja biblioteki dostępna na:
// http://connect.creativelabs.com/openal/Documentation/Forms/AllItems.aspx

// kompilacja: gcc -lopenal -lalut openal.c

const int ILOSC_ZRODEL = 2;
typedef struct {
 const ALbyte* plik;
 ALfloat pitch;
 ALfloat gain;
 ALfloat pozycja[3];
 ALfloat kierunek[3];
 ALuint czy_zapetlone;
} zrodlo_desc;

zrodlo_desc OPIS_ZRODEL[2] = {
 {
  "tlo.wav",
  1.0,
  1.0,
  {0.0,0.0,0.0},
  {0.0,0.0,0.0},
  AL_TRUE
},
 {
  "dzwiek.wav",
  1.0,
  1.0,
  {0.0,0.0,0.0},
  {0.0,0.0,0.0},
  AL_FALSE
}
};

typedef struct {
 ALfloat pozycja[3];
 ALfloat kierunek[3];
 ALfloat orientacja[6];
} sluchacz_desc;

sluchacz_desc OPIS_SLUCHACZA = {
 {0, 0, 0},
 {0, 0, 0},
 {0, 0, 0, 0, 0, 0}
};
```

```
int main(int argc, char *argv[]) {
 alutInit(NULL, 0);
 if(alGetError() != AL_NO_ERROR) {
  puts("Nie można zainicjalizować biblioteki ALUT");
  exit;
 }

/// utworzenie buforów dzwiękowych
 ALuint bufory_dzwiekowe[ILOSC_ZRODEL];
 alGenBuffers(ILOSC_ZRODEL, bufory_dzwiekowe);
 if(alGetError() != AL_NO_ERROR) {
  puts("Nie można zainicjalizować buforów OpenAL");
  exit;
 }

/// utworzenie źródeł dźwiękowych
 ALuint zrodla_dzwiekowe[ILOSC_ZRODEL];
 alGenSources(ILOSC_ZRODEL, zrodla_dzwiekowe);

int i;
 for (i=0; i<ILOSC_ZRODEL; i++) {
  /// ladowanie buforu
  ALenum format;
  ALsizei size;
  ALvoid* data;
  ALsizei freq;
  ALboolean loop;

  alutLoadWAVFile(OPIS_ZRODEL[i].plik, &format, &data, &size, &freq, &loop);
  alBufferData(bufory_dzwiekowe[i], format, data, size, freq);
  alutUnloadWAV(format, data, size, freq);
  // bufory_dzwiekowe[i] = alutCreateBufferFromFile(NAZWY_PLIKOW[i]);

  /// ustawianie parametrów źródła
  // w ogólności ten sam bufor możemy wykorzystać w kilku źródłach
  // ale tutaj tak nie robimy, bo checmy aby każde źródło odgrywało niezależny plik
  alSourcei (zrodla_dzwiekowe[i], AL_BUFFER,   bufory_dzwiekowe[i]);
  alSourcef (zrodla_dzwiekowe[i], AL_PITCH,    OPIS_ZRODEL[i].pitch);
  alSourcef (zrodla_dzwiekowe[i], AL_GAIN,     OPIS_ZRODEL[i].gain);
  alSourcefv(zrodla_dzwiekowe[i], AL_POSITION, OPIS_ZRODEL[i].pozycja);
  alSourcefv(zrodla_dzwiekowe[i], AL_VELOCITY, OPIS_ZRODEL[i].kierunek);
  alSourcei (zrodla_dzwiekowe[i], AL_LOOPING,  OPIS_ZRODEL[i].czy_zapetlone);
 }

/// ustalamy pozycję słuchacza
 alListenerfv(AL_POSITION,    OPIS_SLUCHACZA.pozycja);
 alListenerfv(AL_VELOCITY,    OPIS_SLUCHACZA.kierunek);
 alListenerfv(AL_ORIENTATION, OPIS_SLUCHACZA.orientacja);

// zaczynamy odtwarzać źródło 0
 alSourcePlay(zrodla_dzwiekowe[0]);
 // będzie odtwarzane w pętli dopóki nie zatrzymamy go poprzez
 //    alSourceStop(zrodla_dzwiekowe[0]);
 // lub
 //    alSourcePause(zrodla_dzwiekowe[0]);

/// czekamy chwile, alre dzwiek sie odtwarza
 sleep(10);

i=0;
```

```
 while(i<5) {
  ALint stan_zrodla;
  alGetSourcei(zrodla_dzwiekowe[1], AL_SOURCE_STATE, &stan_zrodla);
  if(stan_zrodla != AL_PLAYING) {
   i++;
   // przemieszczamy źródło w losowe położenie
   OPIS_ZRODEL[1].pozycja[0]=(rand()%100)/10.0;
   OPIS_ZRODEL[1].pozycja[1]=(rand()%100)/10.0;
   OPIS_ZRODEL[1].pozycja[2]=(rand()%100)/10.0;
   alSourcefv(zrodla_dzwiekowe[1], AL_POSITION, OPIS_ZRODEL[1].pozycja);

   // odtwarzamy
   alSourcePlay(zrodla_dzwiekowe[1]);
  }
 }

/// zakanczanie
 alDeleteSources(ILOSC_ZRODEL, zrodla_dzwiekowe);
 alDeleteBuffers(ILOSC_ZRODEL, bufory_dzwiekowe);
 alutExit();
}
```

POKAZ_SLAJDOW_SDL.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje podstawy korzystania z biblioteki SDL:
 * wyświetlanie bitmap, obsługa zdarzeń - klawiatura i tworzenie własnych, timery, ⤸
 czyszczenie ekranu
 *
 * program wyświetla kolejne pliki BMP
 * obrazki zmieniają się co 4 sekundy, możliwe przyspieszenie przez naciśnięcie ⤸
 dowolnego klawisza
 * opuszczenie programu po naciśnięciu "Esc" lub wyświetleniu wszystkich obrazków
 *
 * kompilacja: gcc `sdl-config --cflags --libs` plik_zrodlowy.c -o plik_wynikowy
 *
 * opis API SDL: http://www.libsdl.org/cgi/docwiki.cgi/SDL_20API
 * więcej informacji o SDL po polsku: http://kodowanie.ath.cx/piero/
 *
 */

#include <SDL.h>

void wyswietl_bitmape(char *file, SDL_Surface *ekran) {
 SDL_Surface *image;
 SDL_Rect dest;

// wczytanie obrazka
 image = SDL_LoadBMP(file);
 if ( image == NULL ) {
  fprintf(stderr, "Nie można wczytać %s: %s\n", file, SDL_GetError());
  return;
 }

// ustalenie obszaru na ekranie na ktorym ma byc umieszczony obrazek
 // x,y - gorny lewy rog obrazka, ustawiamy tak aby obrazek byl na srodku ekranu
 dest.x = (ekran->w - image->w)/2;
```

```
 dest.y = (ekran->h - image->h)/2;

// w,h - wysokość i szerokość obrazka
 dest.w = image->w;
 dest.h = image->h;

// skopiowanie obrazka
 SDL_BlitSurface(image, NULL, ekran, &dest);

// aktualizacja wybranego fragmentu ekranu
 SDL_UpdateRects(ekran, 1, &dest);

// usuniecie z pamieci tymczasowego ekranu (obrazka)
 SDL_FreeSurface(image);
}

void wyczysc(SDL_Surface *ekran, SDL_Surface *pusty, SDL_Rect *dest) {
 // skopiowanie pustego ekranu
 SDL_BlitSurface(pusty, NULL, ekran, dest);

// aktualizacja wybranego fragmentu ekranu
 SDL_UpdateRects(ekran, 1, dest);
}

Uint32 wyslij_zdarzenie(Uint32 interval, void *param) {
 SDL_PushEvent(param);
 return interval;
}

int main() {
 // inicjalizacja bliblioteki SDL
 if (SDL_Init(SDL_INIT_VIDEO|SDL_INIT_TIMER|SDL_INIT_EVENTTHREAD)!=0) {
  printf("Błąd inicjalizacji SDL: %s/n",SDL_GetError());
  return -1;
 }


// tworzymy ekran
 SDL_Surface *screen;
 screen = SDL_SetVideoMode(1024,768,24,SDL_HWSURFACE|SDL_FULLSCREEN);
 // vesafb wymaga SDL_SWSURFACE zamiast SDL_HWSURFACE, dziala np. dla
 //    screen = SDL_SetVideoMode(1024,768,24,SDL_SWSURFACE|SDL_FULLSCREEN);
 // zobacz tez SdlVideoStart() w sdl.c w openttd

// tworzymy pusty ekran - przydatny do czyszczenia
 SDL_Surface *black = SDL_CreateRGBSurface(SDL_SWSURFACE, screen->w, screen->h, 24,
  0xff000000, 0x00ff0000, 0x0000ff00, 0x00000000);
 SDL_Rect black_dest; black_dest.x=0; black_dest.y=0; black_dest.w=screen->w; black_dest.↩
 h=screen->h;


// tworzymy unie przechowujaca informacje o zdarzeniach
 SDL_Event zdarzenie_in, zdarzenie_out;

// tworzymy zdarzenie typu nacisniecie klawisza
 zdarzenie_out.type = SDL_KEYDOWN;

// pozbywamy się wszystkich istniejących zdarzeń
 while(SDL_PollEvent(&zdarzenie_in));
```

```
// ustawiamy timer generujący zdarzenie_out co 4s
 SDL_TimerID timer = SDL_AddTimer(4000, wyslij_zdarzenie, &zdarzenie_out);


// tablica wskaźników do ścieżek określających położenia obrazków
 char *nazwy[4];
 nazwy[0] = "./obraz0.bmp";
 nazwy[1] = "./obraz1.bmp";
 nazwy[2] = "./obraz2.bmp";
 nazwy[3] = "./obraz3.bmp";

int i;
 for (i=0; i<4; i++) {
   // wyświetlamy obrazek
   wyswietl_bitmape(nazwy[i], screen);

   // czekamy na zdarzenie - nacisniecie klawisza
   do {
    SDL_WaitEvent(&zdarzenie_in);
   } while (zdarzenie_in.type != SDL_KEYDOWN);
   // moglibyśmy też czekać określoną ilość czasu (np. 4s): SDL_Delay(4000);

   // gdy nacisniety zostal Esc przechodzimy do końca programu ...
   if (zdarzenie_in.key.keysym.sym == SDLK_ESCAPE)
    goto koniec;  // stosowanie instrukcji goto jest niezalecane,
    // jednak w tym wypadku uzasadnione (możnaby jednak używać atexit i exit)

   // czyszczenie ekranu
   wyczysc(screen, black, &black_dest);
 }

koniec:
 // zakończenie pracy biblioteki SDL
 SDL_Quit();
 return 0;
}
```

RYSOWNIK_SDL.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje podstawy korzystania z biblioteki SDL:
 * rysowanie pikseli, obsługa zdarzeń - klawiatura i mysz, zapis obrazu do bitmapy
 *
 * program pozwala rysować kolorem zielonym na czarnym ekranie (ruszanie myszką z ↩
 wciśniętym lewym klawiszem)
 * możliwe jest zapisanie obrazka (pod przypadkową nazwą) przez wcisnięcie klawisza ”S”
 * opuszczenie programu po naciśnięciu ”Esc”
 *
 * kompilacja: gcc ‘sdl-config –cflags –libs‘ plik_zrodlowy.c -o plik_wynikowy
 *
 * opis API SDL: http://www.libsdl.org/cgi/docwiki.cgi/SDL_20API
 * więcej informacji o SDL po polsku: http://kodowanie.ath.cx/piero/
 *
 */

#include <SDL.h>
```

```
void namaluj_pixel(SDL_Surface *ekran, int x, int y, Uint8 R, Uint8 G, Uint8 B){

// tworzymy wskaznik do zadanego piksela
 Uint8 *p = (Uint8 *)ekran->pixels + y * ekran->pitch + x * ekran->format->BytesPerPixel;

// obsługujemy wpisywanie barw w standardach RGB i BGR
 if(SDL_BYTEORDER == SDL_BIG_ENDIAN) {
  p[0] = R; p[1] = G; p[2] = B;
 } else {
  p[0] = B; p[1] = G; p[2] = R;
 }

// aktualizyjemy punkt na ekranie
 SDL_UpdateRect(ekran, x, y, 1, 1);
}

int main() {
 // inicjalizacja bliblioteki SDL
 if (SDL_Init(SDL_INIT_VIDEO|SDL_INIT_TIMER|SDL_INIT_EVENTTHREAD)!=0) {
  printf("Błąd inicjalizacji SDL: %s/n",SDL_GetError());
  return -1;
 }


// tworzymy ekran
 SDL_Surface *screen;
 screen = SDL_SetVideoMode(1024,768,24,SDL_HWSURFACE|SDL_FULLSCREEN);
 // vesafb wymaga SDL_SWSURFACE zamiast SDL_HWSURFACE, dziala np. dla
 //    screen = SDL_SetVideoMode(1024,768,24,SDL_SWSURFACE|SDL_FULLSCREEN);
 // zobacz tez SdlVideoStart() w sdl.c w openttd

// tworzymy unie przechowujaca informacje o zdarzeniach
 SDL_Event zdarzenie;

// pozbywamy się wszystkich istniejących zdarzeń
 while(SDL_PollEvent(&zdarzenie));



// w nieskonczonej pętli czekamy na zdarzenia
 while (1) {
  SDL_WaitEvent(&zdarzenie);
  switch (zdarzenie.type) {
   case SDL_MOUSEMOTION: // ruch myszka
    if ( SDL_GetMouseState(NULL, NULL) & SDL_BUTTON(1) )
     // gdy wciśnięty lewy klawisz
     namaluj_pixel(screen, zdarzenie.motion.x, zdarzenie.motion.y, 0, 255, 0);
     // malujemy na zielono
    break;
   case SDL_KEYDOWN:
    // nacisniecie Esc = wyjscie z programu
    if (zdarzenie.key.keysym.sym == SDLK_ESCAPE)
     goto koniec;
    // naciśnięcie S = zapisanie ekranu jako BMP z unikalną nazwą ...
    else if (zdarzenie.key.keysym.sym == SDLK_s)
     SDL_SaveBMP(screen, tempnam(".", ""));
    break;
  }
 }
```

```
koniec:
 // zakończenie pracy biblioteki SDL
 SDL_Quit();
 return 0;
}
```

OBRAZ_KONTROLNY.CPP

```
/** Program generuje obraz kontrolny z informacją o czasie w oparciu o dostarczona ”tapete”
 * (np. http://commons.wikimedia.org/wiki/Image:Philips_PM5544.svg przekonwertowany do bmp)
 *
 * program może korzystać z libvlc (invmem) celem przesyłania tak generowanego ”programu ⤸
 TV”
 * poprzez sieć lub wyświetlania go poprzez VLC
 *
 * kompilacja: g++ -lSDL -lSDL_ttf -lSDL_gfx -lvlc plik_zrodlowy.c -o plik_wynikowy
 */

#include <SDL/SDL.h>
#include <SDL/SDL_ttf.h>
#include <SDL/SDL_gfxPrimitives.h>
#include <SDL/SDL_rotozoom.h>
#include <SDL/SDL_mutex.h>
#include <time.h>
#include <math.h>
#include <unistd.h>

#define DEBUG(a) printf("%s:%d debug: %s\n", __FILE__, __LINE__, (a))

// stale odpowiadajace polozenim napisu
#define WN_MODE_LEFTUP 1
#define WN_MODE_CENTER 2

// ustawic gdy wyjsciem jest libvlc (moduł invmem
// - http://git.videolan.org/?p=vlc.git;a=blob_plain;f=modules/codec/invmem.c;hb=HEAD)
// w przeciwnym radzie wyjściem będzie SDL
#define VLC


#ifdef VLC
#include <vlc/vlc.h>
#endif

class komunikat {
 private:
  SDL_Surface *napis_caly;
  SDL_Rect dest;
  int x;
  char x_is_not_set;

public:
  int speed;
  int odstep;

  // konstruktory
  komunikat(char* str, char* font_path, int font_size, SDL_Color &font_color, int y) {
   // dla zgodności z wywołaniem funkcji SDLowych kolor odbierany przez referencję
```

```
 //   czyli pobieramy adres zmiennej podanej jako argument
 //   i posługujemy się zmienną o takim adresie,
 //     odpowiada to równoczesnemu przekazywaniu do funkcji wskaźnika na zmienną (&zm),
 //     odbieraniu wskaźnika (*zm) i operowaniu na wyłuskaniu wartości (*zm = ...)
 ustaw_komunikat(str, font_path, font_size, font_color, y);
 speed = 5;
 odstep = 150;
}

// ustawianie punktu startu napisu
void ustaw_start (int x_init) {
 x = x_init;
 x_is_not_set = 0;
}

// ustawianie treści napisu
void ustaw_komunikat (char *str, char *font_path, int font_size, SDL_Color &font_color, int ↩
y) {
 TTF_Font *ttf = TTF_OpenFont(font_path, font_size);
 if (ttf == NULL) {
  printf("Błąd otwarcia czcionki: %s/n",TTF_GetError());
  SDL_Quit();
  return;
 }

 if(!(napis_caly = TTF_RenderUTF8_Solid(ttf, str, font_color))) {
  printf("Błąd renderowania napisu: %s/n",TTF_GetError());
  SDL_Quit();
  return;
 }
 TTF_CloseFont(ttf);

 x_is_not_set=1;
 dest.y=y;
 dest.w=napis_caly->w;
 dest.h=napis_caly->h;
}

// wyświetlanie napisu
void move_and_put(SDL_Surface *ekran) {
 /// gdy x nie był ustawiony to ustawiamy na prawy róg ekranu
 if (x_is_not_set)
  ustaw_start(ekran->w);

 /// przesówamy napis w lewo zgodnie z zadaną prędkoscią
 x -= speed;

 /// jeżeli cały napis nam uciekł za ekran przechodzimy do następnego napisu
 if (x + napis_caly->w < 0)
  x = x + napis_caly->w + odstep;

 /// wyświetlamy półprzezroczysty pasek
 // SDL_SetAlpha(ekran, SDL_SRCALPHA, 125) - ustawia polprzezroczystosc calego surface
 // SDL_SetColorKey - ustawia kolor RGB który ma być uważany za przeźroczysty
 // ale SDL_FillRect z ustawionym kanelem alfa nie powoduje blendingu
 // zatem zkorzystam z funkcji z SDL_gfx:
 //boxRGBA(ekran, , 0, dest->y + napis->w, , 0, 0, 0, 125);
 boxRGBA(ekran, 0, dest.y - 5, ekran->w, dest.y + napis_caly->h + 5, 0, 0, 0, 125);

 /// wypisywanie komunikatu w pętli
```

```
    dest.x = x;
    int i = 1;
    do {
     SDL_BlitSurface(napis_caly, NULL, ekran, &dest);
     SDL_UpdateRects(ekran, 1, &dest);
     dest.x = x + (napis_caly->w + odstep) * i++;
    } while (dest.x < ekran->w);
  }
};

void print_image(SDL_Surface *image, SDL_Surface *ekran) {
 SDL_Rect dest;

// ustalenie obszaru na ekranie na ktorym ma byc umieszczony obrazek
 // x,y - gorny lewy rog obrazka, ustawiamy tak aby obrazek byl na srodku ekranu
 dest.x = (ekran->w - image->w)/2;
 dest.y = (ekran->h - image->h)/2;

// w,h - wysokość i szerokość obrazka
 dest.w = image->w;
 dest.h = image->h;

// skopiowanie obrazka
 SDL_BlitSurface(image, NULL, ekran, &dest);

// aktualizacja wybranego fragmentu ekranu
 SDL_UpdateRects(ekran, 1, &dest);
}

void wyswietl_napis(char *str, SDL_Surface *ekran, int x, int y, char mode,
  TTF_Font *ttf, SDL_Color &kolor_napisu, SDL_Color &kolor_tla) {
 SDL_Rect dest;
 SDL_Surface *napis;

if(!(napis = TTF_RenderUTF8_Shaded(ttf, str, kolor_napisu, kolor_tla))) {
  printf("Błąd renderowania napisu: %s/n",TTF_GetError());
  SDL_Quit();
  return;
 }

// rozmiary napis
 dest.w=napis->w;
 dest.h=napis->h;

// położenie napisu (uwaga napis zawsze umieszczany od lewego górnego rogu dest)
 if (mode == WN_MODE_LEFTUP) {
  dest.x=x;
  dest.y=y;
 } else if (mode == WN_MODE_CENTER) {
  dest.x=x - napis->w /2;
  dest.y=y - napis->h /2;
 }

SDL_BlitSurface(napis, NULL, ekran, &dest);
 SDL_UpdateRects(ekran, 1, &dest);

SDL_FreeSurface(napis);
}

double oblicz_zoom(SDL_Surface *ekran, SDL_Surface *obrazek) {
```

```
 double zoom_x = (double)ekran->w / (double)obrazek->w;
 double zoom_y = (double)ekran->h / (double)obrazek->h;
 if (zoom_y < zoom_x)
  return zoom_y;
 else
  return zoom_x;
}

#ifdef VLC
struct out_vlc {
 SDL_Surface *image;
 SDL_mutex *mutex;
 time_t timestamp;
};

static void * lock(struct out_vlc *out_vlc) {
 SDL_LockMutex(out_vlc->mutex);
 SDL_LockSurface(out_vlc->image);

return out_vlc->image->pixels;
}

static void unlock(struct out_vlc *out_vlc) {
 SDL_UnlockSurface(out_vlc->image);
 SDL_UnlockMutex(out_vlc->mutex);
}
#endif

int main(int argc, char **argv) {
 // inicjalizacja bliblioteki SDL
 if (SDL_Init(SDL_INIT_VIDEO|SDL_INIT_TIMER|SDL_INIT_EVENTTHREAD)!=0) {
  printf("Błąd inicjalizacji SDL: %s/n",SDL_GetError());
  return -1;
 }

// tworzymy unie przechowujaca informacje o zdarzeniach
 SDL_Event zdarzenie;

// inicjalizacja systemu czionek
 if (TTF_Init()) {
  printf("Błąd inicjalizacji SDL_ttf: %s/n",TTF_GetError());
  SDL_Quit();
  return -1;
 }

// wczytany ekran obrazka
 SDL_Surface *obrazek = SDL_LoadBMP("./Philips_PM5544.bmp");
 SDL_Surface *output_image = SDL_LoadBMP("./Philips_PM5544.bmp");
  // zamiast SDL_CreateRGBSurface z którą były problemy
 if ( obrazek == NULL || output_image == NULL ) {
  fprintf(stderr, "Nie można wczytać obrazka: %s\n", SDL_GetError());
  return -1;
 }

// ładowanie czionek
 TTF_Font *ttf_name = TTF_OpenFont(
  "/usr/share/fonts/truetype/ttf-dejavu/DejaVuSansMono-Bold.ttf",
  30
 );
 TTF_Font *ttf_time = TTF_OpenFont(
```

```
  "/usr/share/fonts/truetype/ttf-dejavu/DejaVuSansMono-Bold.ttf",
  30
 );
 if (ttf_name == NULL || ttf_time == NULL) {
  printf("Błąd otwarcia czcionki: %s/n",TTF_GetError());
  SDL_Quit();
  return -1;
 }

double zoom = 0.35;
 int argc2 = argc; // zmienna pomocnicza przechowująca liczbę nieobsłużonych argumentów
#ifdef VLC
 struct out_vlc out_vlc;
 out_vlc.mutex = SDL_CreateMutex();
 out_vlc.image = SDL_CreateRGBSurface(SDL_SWSURFACE, zoom*obrazek->w, zoom*obrazek->h, 24, 0, ⤸
 0, 0, 0);

char clock[64], cunlock[64], cdata[64], width[32], height[32];
 libvlc_exception_t ex;
 libvlc_instance_t *libvlc;
 libvlc_media_t *m;
 libvlc_media_player_t *mp;
 char const *vlc_argv[] =
 {
  //"-q",
  // "-vvvvv",
  // "--plugin-path", "/sciezka/do/katalogu/DODATKOWYCH/modulow/VLC",
  "--ignore-config", // Don't use VLC's config files
  "--noaudio",
  "--codec", "invmem",
  //"--quiet-synchro",
  //"--aout", "alsa",
  "--fake-fps", "10",
  "--invmem-width", width,
  "--invmem-height", height,
  "--invmem-lock", clock,
  "--invmem-unlock", cunlock,
  "--invmem-data", cdata,
 };
 int vlc_argc = sizeof(vlc_argv) / sizeof(*vlc_argv);

sprintf(clock, "%lld", (long long int)(intptr_t)lock);
 sprintf(cunlock, "%lld", (long long int)(intptr_t)unlock);
 sprintf(cdata, "%lld", (long long int)(intptr_t)&out_vlc);
 sprintf(width, "%i", out_vlc.image->w);
 sprintf(height, "%i", out_vlc.image->h);

libvlc_exception_init(&ex);
 libvlc = libvlc_new(vlc_argc, vlc_argv, &ex);
 m = libvlc_media_new(libvlc, "fake://", &ex);
 mp = libvlc_media_player_new_from_media(m, &ex);
 libvlc_media_release(m);
 libvlc_media_player_play(mp, &ex);

SDL_Surface *screen = SDL_SetVideoMode(320,240,24,SDL_HWSURFACE|SDL_RESIZABLE);
#else
 // tworzymy ekran
 SDL_Surface *screen;
 if ( argc >= 2 && ! strcmp(argv[1], "-w") ) {
  // okienko
```

```
   screen = SDL_SetVideoMode(320,240,24,SDL_HWSURFACE|SDL_RESIZABLE);
   argc2--;
  } else {
   // pełny ekran
   screen = SDL_SetVideoMode(1024,768,24,SDL_HWSURFACE|SDL_FULLSCREEN);
  }

 // ustalenie skalowania
  zoom = oblicz_zoom(screen, obrazek);
#endif

 // kolory napisów
  SDL_Color kolor_tla={0,0,0}, kolor_napisu={0xff,0xff,0xff};

 // umieszczenie nazwy na obrazku
  wyswietl_napis((char*)"www.opcode.eu.org", obrazek,
   512 * obrazek->w / 1025, 623 * obrazek->h / 770,
   WN_MODE_CENTER, ttf_name, kolor_napisu, kolor_tla);

 // tworzymy komunikat specjalny
  komunikat * napis_specjalny = NULL;
  if (argc2 == 2)
   napis_specjalny = new komunikat(
    argv[argc - 1],
    (char*)"/usr/share/fonts/truetype/ttf-dejavu/DejaVuSansMono-Bold.ttf",
    32,
    kolor_napisu,
    715 * obrazek->h / 770
   ); // komunikat tworzymy w oparciu o ostatni z argumentów

 // położenia zegarów
  const int time1_x = 311 * obrazek->w / 1025;
  const int time2_x = 713 * obrazek->w / 1025;
  const int time_y = 393 * obrazek->h / 770;

 // zmienne pomocnicze
  time_t czas;
  struct tm czas_rozlozony;
  char czas_napis[100];
  int i=0;
  SDL_Surface *scaled_image;

 while (1) {
   // kopiujemy tło na ekran roboczy
   print_image(obrazek, output_image);

   // odczytujemy czas
   time(&czas);
   gmtime_r(&czas, &czas_rozlozony); // localtime_r(&czas, &czas_rozlozony);

   // tworzymy napis i dodajemy go do ekranu roboczego
   strftime(czas_napis, sizeof(czas_napis), "%Y-%m-%d", &czas_rozlozony);
   wyswietl_napis(czas_napis, output_image, time1_x, time_y,
    WN_MODE_CENTER, ttf_time, kolor_napisu, kolor_tla);
   strftime(czas_napis, sizeof(czas_napis), "%H:%M:%S UTC", &czas_rozlozony);
   wyswietl_napis(czas_napis, output_image, time2_x, time_y,
    WN_MODE_CENTER, ttf_time, kolor_napisu, kolor_tla);

   if (napis_specjalny != NULL)
    napis_specjalny->move_and_put(output_image);
```

```
  // skalujemy
  if (zoom != 1) {
   scaled_image = zoomSurface(output_image, zoom, zoom, 1);
  } else {
   scaled_image = output_image;
  }

  // przenosimy ekran roboczy na ekran docelowy
#ifdef VLC
  SDL_LockMutex(out_vlc.mutex);
  print_image(scaled_image, out_vlc.image);
  out_vlc.timestamp = czas;
  SDL_UnlockMutex(out_vlc.mutex);
  SDL_SaveBMP(scaled_image, "out.bmp");
#else
  print_image(scaled_image, screen);
#endif

  if (zoom != 1) {
   SDL_FreeSurface(scaled_image);
  }

  // obsługujemy ewentualne zdarzenia
  // warto zauważyć że korzystamy z nieblokującej funkcji SDL_PollEvent zamiast ↩
  SDL_WaitEvent
  while (SDL_PollEvent(&zdarzenie)) {
   if (zdarzenie.type == SDL_QUIT || zdarzenie.type == SDL_KEYDOWN && zdarzenie.key.keysym.↩
   sym == SDLK_ESCAPE) {
    libvlc_media_player_stop(mp, &ex);
    libvlc_media_player_release(mp);
    libvlc_release(libvlc);
    SDL_DestroyMutex(out_vlc.mutex);
    SDL_FreeSurface(out_vlc.image);
    SDL_FreeSurface(output_image);
    SDL_Quit();
    exit (0);
#ifndef VLC
   } else if (zdarzenie.type ==  SDL_VIDEORESIZE) {
    screen = SDL_SetVideoMode(zdarzenie.resize.w, zdarzenie.resize.h,
     24, SDL_HWSURFACE|SDL_RESIZABLE);
    zoom = oblicz_zoom(screen, obrazek);
#endif
   }
  }
  // czekamy 0.1s
  SDL_Delay(100);
 }
}
```

READ_PNG.C

```
/*
 * plik ma na celu prezentację podstawowych sposobu czytania plików png z
 * wykorzystaniem biblioteki libpng
 */

#include <png.h>
```

```c
#define E_FOPEN       -1
#define E_READ_STRUCT -2
#define E_INFO_STRUCT -3
#define E_PNG_TYPE    -4

int read_png(char *file_name) {
 png_structp png_ptr;
 png_infop info_ptr;
 png_uint_32 width, height;
 int bit_depth, color_type, interlace_type;
 int col, row, row_cnt, skip_cnt, row_len;
 FILE *fp;

fp = fopen(file_name, "rb");
 if (fp == NULL) {
  return E_FOPEN;
 }

png_ptr = png_create_read_struct(PNG_LIBPNG_VER_STRING, NULL, NULL, NULL);
 if (png_ptr == NULL) {
  fclose(fp);
  return E_READ_STRUCT;
 }

info_ptr = png_create_info_struct(png_ptr);
 if (info_ptr == NULL || setjmp(png_jmpbuf(png_ptr))) {
  fclose(fp);
  png_destroy_read_struct(&png_ptr, png_infopp_NULL, png_infopp_NULL);
  return E_INFO_STRUCT;
 }

png_init_io(png_ptr, fp);

png_read_png(
  png_ptr, info_ptr,
  PNG_TRANSFORM_STRIP_16 | PNG_TRANSFORM_STRIP_ALPHA | PNG_TRANSFORM_PACKING | ⤸
  PNG_TRANSFORM_EXPAND, NULL
);

png_get_IHDR(png_ptr, info_ptr, &width, &height, &bit_depth, &color_type, &interlace_type, ⤸
 NULL, NULL);

// w zależności od typu pliku png ustalamy co ile bajtów jest interesująca nas ⤸
 informacja
 if (color_type == PNG_COLOR_TYPE_GRAY) {
  puts("GRAY");
  skip_cnt = 1;
 } else if (color_type == PNG_COLOR_TYPE_RGB) {
  puts("RGB");
  skip_cnt = 3;
 } else {
  fclose(fp);
  png_destroy_read_struct(&png_ptr, png_infopp_NULL, png_infopp_NULL);
  return E_PNG_TYPE;
 }
 row_len = width * skip_cnt;

// czytamy kolejne wiersze (jedno-pixelowe linie) z pliku
 png_bytepp row_pointers = png_get_rows(png_ptr, info_ptr);
```

```
 for (row=0; row<height; row++) {
  row_cnt = row * row_len;
  for (col=0; col<row_len; col=col+skip_cnt) {
   if (row_pointers[row][col] < 50)
    printf("*");
   else
    printf(" ");
  }
  printf("\n");
 }

png_destroy_read_struct(&png_ptr, &info_ptr, png_infopp_NULL);
 fclose(fp);
 return 0;
}

main() {
 printf("Return %d\n", read_png("test.png"));
}
```

OPENGL-MAIN-GLUT_SDL_GLX.C

```
/**

plik odpowiedzialny za obsługę "bibliotek wysokopoziomowych" OpenGL - glut, sdl, glx

pokazuje różnice w inicjalizacji poszczególnych bibliotek, a także możliwość ich ⤸
mieszania
(np. SDL wraz z swoimi udogodnieniami jak timery itp + GLUT z swoimi funkcji do obsługi ⤸
napisów itp)

OpenGL jest biblioteką (API) do obsługi grafiki 3D, potrafi on wykorzystywać wsparcie ⤸
sprzętowe
GPU zgodnych z tą specyfikacją (oprócz tego jest implementacja programowa - MESA).
Biblioteka stworzona jest w architekturze klient-serwer. Klientem jest program zlecający ⤸
operacje 3D,
a serwerem implementacja OpenGL (programowa lub korzystająca z wsparcia sprzętowego - w ⤸
sterowniku
karty graficznej). W szczególności biblioteka pozwala na użycie zdalnego serwera.

Pierwszą rzeczą, która rzuca się w oczy przy korzystaniu z OpenGL jest to iż jest on ⤸
maszyną stanu.
Korzystanie z tej biblioteki polega w dużej mierze na wywoływaniu funkcji odpowiedzialnych
za zmiany stanu w jakim pracuje serwer OpenGL (np. funkcje glEnable, funkcje ustawiające
parametry renderingu itp).

Część funkcjonalności OpenGL jest implementowana w postaci opcjonalnych rozszerzeń
(wiele z nich wchodzi potem do kolejnego wydania podstawowej specyfikacji),
wsparcie w rozpoznawaniu rozszerzeń obsługiwanych przez aktualną implementację
zapewnia biblioteka GLEW.

Oprócz samego OpenGL funkcjonują biblioteki ułatwiające korzystanie z niego (tworzenie ⤸
okna itp)
i dodające m.in. funkcjonalności obsługi wejścia takie jak porównywane w tym pliku GLUT, ⤸
SDL, GLX.

Ogólnie zauważalna jest tendencja do coraz swobodniejszego programowania GPU - zauważalne
jest to chociażby w zwiększaniu roli programowalnego potoku (shadery itp), a minimalizacji
```

znaczenia potoku stałego (patrz np. OpenGL ES 2.0). Rozwój GPU i zwiększanie ilości pamięci
na kartach zaczyna pociągać za sobą zagadnienie zarządzania tą pamięcią przez sterownik
karty / system operacyjny (pewne echa tego będą wspomniane w jednym z przykładowych programów)
Przy omawianiu tego typu zagadnień warto wspomnieć o OpenCL - języku stworzonym z myślą o
możliwości swobodnego przerzucenia obliczeń pomiędzy CPU a GPU (obecnie moce obliczeniowe
GPU w niektórych zagadnieniach znacznie przekraczają moce obliczeniowe CPU - patrz GPGPU
czyli General Purpose Graphics Processing Unit).

Zobacz też:
* http://nehe.gamedev.net/
* http://www.dimmension3.spine.pl/modules.php?name=Tutorials&wtd=show_cat&cat=opengl
* http://januszg.hg.pl/opengl/index.html

**/

```
/// numeryczne identyfikatory trybów kompilacji
#define GLUT 1
// gcc -lGL -lGLU -lglut -D MODE=1 opengl-main-glut_sdl_glx.c && ./a.out
#define SDL 2
// gcc -lGL -lGLU -lSDL -D MODE=2 opengl-main-glut_sdl_glx.c && ./a.out
#define GLX 3
// gcc -lGL -lGLU -lXxf86vm -D MODE=3 opengl-main-glut_sdl_glx.c && ./a.out

/// określamy plik z funkcją używaną do rysowania
#ifndef DRAW_FILE
# define DRAW_FILE "opengl-draw-simple.c"
#endif

/// włączamy plik odpowiedzialny za ustawienia "czystego" OpenGL
#include "opengl-init.c"

/// standardowe pliki nagłówkowe
#include <stdio.h>
#include <stdlib.h> // exit
#include <unistd.h> // sleep, usleep


/// funkcja główna (w trzech wariantach) - przygotowanie do funkcji obsługi OpenGL
/// oraz funkcje pomnocnicze dla niej
/// i nagłówki odpowiednich bibliotek "wysokopoziomowych" OpenGL
#if MODE == GLUT
 /// GLUT (-lglut)
 #include <GL/glut.h>

// określenie funkcji rysującej na ekran (zależne od MODE)
 #define DRAW_TO_SCREEN glutSwapBuffers()
 // makro określające że wykonana była inicjalizacja glut'a
 // i wolno korzystać z jego funkcji
 #define GLUT_INIT 1
 // włączenie pliku obsługującego rysowanie "czystym" OpenGL
 #include DRAW_FILE

// globalna zmienna z identyfikatorem okna GLUT'owego
 int okno;

// funkcja obsługująca reakcję na zdarzenia klawiatury
```

```
void nacisnieto_klawisz(unsigned char klawisz, int x, int y) {
  // naciśnięcie q lub ESCAPE (kod ASCII)
  // klawisze specjalne zdefiniowane w GL/glut.h (np. GLUT_KEY_DOWN)
  switch (klawisz) {
   case 'q':
   case 0x1b:
     // niszczymy okno GLUT'owe
     glutDestroyWindow(okno);
     // kończymy program
     exit(0);
   case 'F':
     // przejście w tryb pełnoekranowy
     glutFullScreen();
     // w zasadzie przedtem powinniśmy zapamiętać:
     // glutGet(GLUT_WINDOW_WIDTH) glutGet(GLUT_WINDOW_HEIGHT)
     // oraz położenie okna aby muc wrócić do trybu okienkowego przy pomocy:
     // glutReshapeWindow() i glutPositionWindow()
     break;
  }
 }

// funkcja główna
 int main (int argc, char **argv) {
  // inicjalizacja GLUTa i przekazanie do niego argumentów wywołania
  glutInit(&argc, argv);
  // ustawienie trybu wyświetlania
  glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE | GLUT_ALPHA | GLUT_DEPTH);
  // ustawienie początkowego rozmiaru okna GLUTowego
  glutInitWindowSize(500, 300);
  // ustawienie położenia okna GLUTowego (lewy górny róg)
  glutInitWindowPosition(0, 0);
  // utworzenie okna
  okno = glutCreateWindow("tytul okna GLUT");

  // przejście w tryb pełnoekranowy
  //glutFullScreen();

  // określenie funkcji odpowiedzialnej za rysowanie sceny
  glutDisplayFunc(&narysuj_scene);
  // określenie funkcji wykonywanej gdy nie ma zdarzeń do obsłużenia
  glutIdleFunc(&narysuj_scene);
  // okręslenie funkcji wykonywane przy zmianie rozmiaru okna
  glutReshapeFunc(&ustaw_rozmiar_sceny);

  // określenie funkcji wywoływanej po naciśnięciu przycisku na klawiaturze
  glutKeyboardFunc(&nacisnieto_klawisz);

  // inicjalizacja OpenGL
  inicjalizuj_GL();

  // przygotowanie obiektów dla funkcji rysującej
  przygotuj_scene();

  // uruchomienie pętli przetwarzającej zdarzenia GLUT
  glutMainLoop();
 }
#elif MODE == SDL
 /// SDL (-lSDL)
 #include <SDL/SDL.h>
 #ifdef GLUT_INIT
```

```
  #include <GL/glut.h> // GLUT (-lglut)
 #endif
 // określenie funkcji rysującej na ekran (zależne od MODE)
 #define DRAW_TO_SCREEN SDL_GL_SwapBuffers()
 // włączenie pliku obsługującego "czysty" OpenGL
 #include DRAW_FILE

// zmienna globalna będąca wskaźnikiem na powierzchnię rysunkową
 SDL_Surface *powierzchnia;

int main (int argc, char **argv) {
  // pomimo używania innej biblioteki możemy korzystać z
  // pewnych funkcji GLUTa (np. umieszczających napisy czy też obiekty geometryczne)
  // jeżeli dokonamy jego inicjalizacji
  #ifdef GLUT_INIT
   glutInit(&argc, argv);
  #endif

  // inicjalizacja bliblioteki SDL
  if (SDL_Init(SDL_INIT_VIDEO|SDL_INIT_TIMER|SDL_INIT_EVENTTHREAD)!=0) {
   printf("Błąd inicjalizacji SDL: %s/n",SDL_GetError());
   SDL_Quit(); // likwidacja okna SDL
   return -1;
  }

  // pobranie informacji o urządzeniu wideo
  const SDL_VideoInfo *video_info;
  if ( (video_info = SDL_GetVideoInfo()) == NULL ) {
   printf("Błąd odczytu informacji o video: %s/n",SDL_GetError());
   SDL_Quit(); // likwidacja okna SDL
   return -1;
  }

  // ustawienie flag dla trybu wideo w zależności od posiadanego sprzętu
  int flagi_trybu_wideo = 0;
  if ( video_info->hw_available ) {
   flagi_trybu_wideo = SDL_HWSURFACE;
   puts("SDL_HWSURFACE - \"powierzchnia\" w pamięci karty graficznej");
  } else {
   flagi_trybu_wideo = SDL_SWSURFACE;
   puts("SDL_SWSURFACE - \"powierzchnia\" w pamięci programowej");
  }
  if ( video_info->blit_hw ) {
   flagi_trybu_wideo |= SDL_HWACCEL;
   puts("SDL_HWACCEL - akceleracja sprzętowa");
  } else {
   puts("NO SDL_HWACCEL - brak akceleracji sprzętowej");
  }
  // rozszerzenie wcześniej ustawionych właściwości trybu widoo o:
  // OpenGL + podwójne buferowanie + sprzętowa paleta + okno z możliwością zamiany
  rozmiaru
  flagi_trybu_wideo |= SDL_OPENGL | SDL_GL_DOUBLEBUFFER | SDL_HWPALETTE | SDL_RESIZABLE;

  // włączamy podwójne buferowanie OpenGL
  SDL_GL_SetAttribute( SDL_GL_DOUBLEBUFFER, 1 );

  // uzyskanie "powierzchni" o zadanym rozmiarze (500x300), głębi kolorów (8 bitów)
  // i właściwościach (ustawione wcześniej flagi)
  if ( (powierzchnia = SDL_SetVideoMode( 500, 300, 8, flagi_trybu_wideo)) == NULL ) {
   printf("Błąd pobrania \"powierzchni\" SDL: %s/n",SDL_GetError());
```

```
    SDL_Quit(); // likwidacja okna SDL
    return -1;
  }

  // inicjalizacja OpenGL
  inicjalizuj_GL();
  // ustawienie rozmiaru sceny
  ustaw_rozmiar_sceny( 500, 300 );

  // przygotowanie obiektów dla funkcji rysującej
  przygotuj_scene();

  // pętla zdarzeń
  SDL_Event zdarzenie;
  char okno_jest_aktywne;
  while (1) {
   while ( SDL_PollEvent( &zdarzenie ) ) {
    switch (zdarzenie.type) {
     case SDL_KEYDOWN:
      // naciśnięty przycisk
      switch ( zdarzenie.key.keysym.sym ) {
       case SDLK_F2:
        // przełączenie w tryb pełnoekranowy
        SDL_WM_ToggleFullScreen( powierzchnia );
        break;
       case SDLK_ESCAPE:
        // likwidacja okna SDL
        SDL_Quit();
        // zakończenie programu
        exit(0);
      }
      break;
     case SDL_VIDEORESIZE:
      // zmienił się rozmiar okna
      if ( (powierzchnia = SDL_SetVideoMode(
       zdarzenie.resize.w, zdarzenie.resize.h, 8, flagi_trybu_wideo)) == NULL ) {
       printf("Błąd pobrania \"powierzchni\" SDL po zmianie rozmiaru okna: %s/n",
        SDL_GetError());
       SDL_Quit(); // likwidacja okna SDL
       return -1;
      }
      ustaw_rozmiar_sceny( zdarzenie.resize.w, zdarzenie.resize.h );
      break;
     case SDL_ACTIVEEVENT:
      // okno zostało aktywowane lub deaktywowane
      okno_jest_aktywne = zdarzenie.active.gain;
      // zapamiętujemy czy okno aktywne
      break;
     case SDL_QUIT:
      SDL_Quit();
      exit (0);
    }
   }
   // rysowanie tylko gdy okno aktywne
   if ( okno_jest_aktywne )
    narysuj_scene();
  }
 }
#elif MODE == GLX
 /// GLX (-)
```

```
#include <GL/glx.h>
#ifdef GLUT_INIT
 #include <GL/glut.h> // GLUT (-lglut)
#endif
// X11 (-lXxf86vm)
#include <X11/extensions/xf86vmode.h>
#include <X11/keysym.h>

// UWAGA: do działania wymagany pakiet libgl1-mesa-glx a nie libgl1-mesa-swx11

// struktura przechowująca informacje o oknie
struct {
 Display *wyswietlacz;
 int ekran;
 Window okno;

 GLXContext kontext_gl;
 XSetWindowAttributes attrybuty;
 XF86VidModeModeInfo tryb_x_serwera;

 char pelny_ekran;
 unsigned int okno_szer, scena_szer;
 unsigned int okno_wys, scena_wys;
} oknoGLX;

// określenie funkcji rysującej na ekran (zależne od MODE)
#define DRAW_TO_SCREEN glXSwapBuffers(oknoGLX.wyswietlacz, oknoGLX.okno)
// włączenie pliku obsługującego "czysty" OpenGL
#include DRAW_FILE

void stworz_okno_glx() {
  // połączenie z Xserwerem
  oknoGLX.wyswietlacz = XOpenDisplay(0);
  oknoGLX.ekran = DefaultScreen(oknoGLX.wyswietlacz);

  // pobieramy informację o trybach ekranu
  XF86VidModeModeInfo **tryby;
  int ile_trybow;
  XF86VidModeGetAllModeLines(oknoGLX.wyswietlacz, oknoGLX.ekran, &ile_trybow, &tryby);
  // zachowujemy informację o obecnie używanym trybie ekranu
  oknoGLX.tryb_x_serwera = *tryby[0];

  // uzyskiwania visuala o zadanych atrybutach
  XVisualInfo *visual;
  int attrList[] = {GLX_RGBA, GLX_RED_SIZE, 4, GLX_GREEN_SIZE, 4, GLX_BLUE_SIZE, 4,
  GLX_DEPTH_SIZE, 16, GLX_DOUBLEBUFFER, None};
  visual = glXChooseVisual(oknoGLX.wyswietlacz, oknoGLX.ekran, attrList);
  if (visual == NULL) {
   // gdy się nie udało próbujemy bez podwójnego buforowania
   attrList[9] = None;
   visual = glXChooseVisual(oknoGLX.wyswietlacz, oknoGLX.ekran, attrList);
  }

  // tworzenie kontekstu GLX
  oknoGLX.kontext_gl = glXCreateContext(oknoGLX.wyswietlacz, visual, 0, GL_TRUE);

  // tworzenie mapy kolorów
  oknoGLX.attrybuty.colormap = XCreateColormap(oknoGLX.wyswietlacz,
   RootWindow(oknoGLX.wyswietlacz, visual->screen), visual->visual, AllocNone);
  oknoGLX.attrybuty.border_pixel = 1;
```

```
oknoGLX.attrybuty.event_mask = ExposureMask | KeyPressMask | ButtonPressMask |
 StructureNotifyMask;

// tworzenie okna w trybie pełnoekranowym
if (oknoGLX.pelny_ekran) {
 // wybieramy tryb ekranu zgodny z naszymi preferencjami
 int i;
 int preferowany_tryb = 0;
 for (i = 0; i < ile_trybow; i++) {
  if ((tryby[i]->hdisplay == oknoGLX.okno_szer)
  && (tryby[i]->vdisplay == oknoGLX.okno_wys)) {
   preferowany_tryb = i;
   break;
  }
 }
 XF86VidModeSwitchToMode(oknoGLX.wyswietlacz, oknoGLX.ekran, tryby[preferowany_tryb]);
 XF86VidModeSetViewPort(oknoGLX.wyswietlacz, oknoGLX.ekran, 0, 0);
 oknoGLX.attrybuty.override_redirect = True;

 oknoGLX.okno = XCreateWindow(oknoGLX.wyswietlacz,
  RootWindow(oknoGLX.wyswietlacz, visual->screen),
  0, 0, tryby[preferowany_tryb]->hdisplay, tryby[preferowany_tryb]->vdisplay, 0,
  visual->depth, InputOutput, visual->visual,
  CWBorderPixel | CWColormap | CWEventMask | CWOverrideRedirect,
  &oknoGLX.attrybuty);

 XWarpPointer(oknoGLX.wyswietlacz, None, oknoGLX.okno, 0, 0, 0, 0, 0, 0);
 XMapRaised(oknoGLX.wyswietlacz, oknoGLX.okno);
 XGrabKeyboard(oknoGLX.wyswietlacz, oknoGLX.okno, True, GrabModeAsync,
  GrabModeAsync, CurrentTime);
 XGrabPointer(oknoGLX.wyswietlacz, oknoGLX.okno, True, ButtonPressMask,
  GrabModeAsync, GrabModeAsync, oknoGLX.okno, None, CurrentTime);

 // dla poprawnego skalowania sceny
 oknoGLX.scena_szer = tryby[preferowany_tryb]->hdisplay;
 oknoGLX.scena_wys = tryby[preferowany_tryb]->vdisplay;
// tworzenie normalnego okna
} else {
 oknoGLX.okno = XCreateWindow(oknoGLX.wyswietlacz,
  RootWindow(oknoGLX.wyswietlacz, visual->screen),
  0, 0, oknoGLX.okno_szer, oknoGLX.okno_wys, 0, visual->depth,
  InputOutput, visual->visual,CWBorderPixel | CWColormap | CWEventMask,
  &oknoGLX.attrybuty);

 Atom wm_delete = XInternAtom(oknoGLX.wyswietlacz, "WM_DELETE_WINDOW", True);
 XSetWMProtocols(oknoGLX.wyswietlacz, oknoGLX.okno, &wm_delete, 1);
 XSetStandardProperties(oknoGLX.wyswietlacz, oknoGLX.okno, "tytul okna GLX",
  "tytul okna GLX", None, NULL, 0, NULL);
 XMapRaised(oknoGLX.wyswietlacz, oknoGLX.okno);

 // dla poprawnego skalowania sceny
 oknoGLX.scena_szer = oknoGLX.okno_szer;
 oknoGLX.scena_wys = oknoGLX.okno_wys;
}
XFree(visual);
XFree(tryby);

// łączenie kontekstu z oknem
glXMakeCurrent(oknoGLX.wyswietlacz, oknoGLX.okno, oknoGLX.kontext_gl);
// XGetGeometry możemy pobrać aktualną geometrię okna
```

```
  // poprzez glXIsDirect możemy sprawdzić czy używamy DRI

  // inicjalizacja OpenGL
  inicjalizuj_GL();
  // ustawienie rozmiaru sceny
  ustaw_rozmiar_sceny(oknoGLX.scena_szer, oknoGLX.scena_wys);
 }

void zniszcz_okno_glx() {
  if (oknoGLX.kontext_gl) {
   // jeżeli był to zwalniamy kontekst
   glXMakeCurrent(oknoGLX.wyswietlacz, None, NULL);
   glXDestroyContext(oknoGLX.wyswietlacz, oknoGLX.kontext_gl);
   oknoGLX.kontext_gl = NULL;
  }
  if (oknoGLX.pelny_ekran) {
   // jeżeli był tryb pełnoekranowy to przywracamy orginalne ustawienia Xserwera
   XF86VidModeSwitchToMode(oknoGLX.wyswietlacz, oknoGLX.ekran, &oknoGLX.tryb_x_serwera);
   XF86VidModeSetViewPort(oknoGLX.wyswietlacz, oknoGLX.ekran, 0, 0);
  }
  // zamykamy połączenie z wyświetlaczem
  XCloseDisplay(oknoGLX.wyswietlacz);
 }

int main (int argc, char **argv) {
  // pomimo używania innej biblioteki możemy korzystać z
  // pewnych funkcji GLUTa (np. umieszczających napisy czy też obiekty geometryczne)
  // jeżeli dokonamy jego inicjalizacji
  #ifdef GLUT_INIT
   glutInit(&argc, argv);
  #endif

  // wstępne ustawienia okna
  oknoGLX.pelny_ekran=0;
  oknoGLX.okno_szer=500;
  oknoGLX.okno_wys=300;

  // tworzymy okno
  stworz_okno_glx();

  // przygotowanie obiektów dla funkcji rysującej
  przygotuj_scene();

  // pętla zdarzeń
  XEvent zdarzenie;
  char okno_jest_aktywne;
  KeySym klawisz;
  while (1) {
   while (XPending(oknoGLX.wyswietlacz) > 0) {
    // jeżeli są oczekujące zdarzenia odbieramy je
    XNextEvent(oknoGLX.wyswietlacz, &zdarzenie);
    switch (zdarzenie.type) {
     // okno zostało pokazane
     case Expose:
      if (zdarzenie.xexpose.count == 0)
       narysuj_scene();
      break;
     // zmiana w konfiguracji okna
     case ConfigureNotify:
      // jeżeli rozmiar uległ rzeczywistej zmianie
```

```
        if ( (zdarzenie.xconfigure.width != oknoGLX.okno_szer) ||
          (zdarzenie.xconfigure.height != oknoGLX.okno_wys) ) {
         oknoGLX.scena_szer = oknoGLX.okno_szer = zdarzenie.xconfigure.width;
         oknoGLX.scena_wys = oknoGLX.okno_wys = zdarzenie.xconfigure.height;
         ustaw_rozmiar_sceny(oknoGLX.okno_szer, oknoGLX.okno_wys);
        }
        break;
       // naciśnięcie przycisku na klawiaturze
       case KeyPress: // ButtonPress -> zdarzenie myszy
        klawisz = XLookupKeysym(&zdarzenie.xkey, 0);
        switch (klawisz) {
         case XK_Escape:
          puts("close via XK_Escape");
          zniszcz_okno_glx();
          exit(0);
          break;
         case XK_F2:
          puts("switch fullscreen");
          zniszcz_okno_glx();
          oknoGLX.pelny_ekran = !oknoGLX.pelny_ekran;
          stworz_okno_glx();
          break;
        }
        break;
       // wiadomość do klienta
       case ClientMessage:
        // obsługa zamknięcia okna poprzez komunikat X'ów
        // (np. nacisnięcie krzyżyka zamykającego na belce tytułowej)
        if (*XGetAtomName(oknoGLX.wyswietlacz, zdarzenie.xclient.message_type) == ⤸
        *"WM_PROTOCOLS") {
         puts("close via WM_PROTOCOLS");
         zniszcz_okno_glx();
         exit(0);
        }
        break;
      }
     }
     narysuj_scene();
    }
   }
#endif
```

OPENGL-MAIN-GLUT.C

```
/**

plik odpowiedzialny za obsługę "bibliotek wysokopoziomowych" OpenGL - glut

plik ten demonstruje bardziej zaawansowane korzystanie z GLUTa
(niepokazane w opengl-main-glut_sdl_glx.c dla innych mechanizmów wysokopoziomowych)

WARTO JESZCZE ZAPOZNAĆ SIĘ Z:
* vertex i pixel (fragment) shadery - OpenGL Shading Language (GLSL) - http://www.⤸
lighthouse3d.com/opengl/glsl/
* cienie - http://forum.gamedev.pl/index.php/topic,754.0.html

* antyaliasing i multisampling,
* efekty realizowane w oparciu o bufory akumulacyjny (rozmycie ruchu itp) i szablonowy (⤸
```

```
lustro)
* tryb selekcji obiektów - obsługa strzału w obiekt
  (tu warto wspomnieć o trzecim trybie - sprzężenia zwrotnego - dającym precyzyjniejsze ⤸
  informacje ...)
* efekty cząsteczkowe (Sprajty punktowe), czyli jak zrobić deszcz
* ewaluacja krzywych i powierzchni

Zobacz też:
* http://nehe.gamedev.net/
* http://www.dimmension3.spine.pl/modules.php?name=Tutorials&wtd=show_cat&cat=opengl
* http://januszg.hg.pl/opengl/index.html

**/

//#define DEBUG(a) puts(a);
#define DEBUG(a) ;

/// ustawiamy MODE == GLUT
#define GLUT 1
#define MODE 1
// gcc -lGL -lGLU -lglut -lglpng   -D DRAW_FILE=\"opengl-draw-simple.c\" opengl-main-glut.c && ⤸
./a.out

/// określamy plik z funkcją używaną do rysowania
#ifndef DRAW_FILE
# define DRAW_FILE "opengl-draw-light.c"
#endif

/// włączamy plik odpowiedzialny za ustawienia "czystego" OpenGL
#include "opengl-init.c"

/// standardowe pliki nagłówkowe
#include <stdio.h>
#include <stdlib.h> // exit
#include <unistd.h> // sleep, usleep


/// funkcja główna - przygotowanie do funkcji obsługi OpenGL
/// oraz funkcje pomnocnicze dla niej
/// GLUT (-lglut)
#include <GL/glut.h>

// określenie funkcji rysującej na ekran (zależne od MODE)
#define DRAW_TO_SCREEN glutSwapBuffers()
// makro określające że wykonana była inicjalizacja glut'a
// i wolno korzystać z jego funkcji
#define GLUT_INIT 1
// włączenie pliku obsługującego rysowanie "czystym" OpenGL
#include DRAW_FILE

// globalna zmienna z identyfikatorem okna GLUT'owego
int okno;

// definicje kroków związanych z modyfikacjami kamery
#define step_xyz 0.05
#define step_obr 0.05
#define step_zoom 0.05
#define max_zoom 20
#define min_zoom 80
```

```
// funkcja obsługująca reakcję na zdarzenia klawiatury
void nacisnieto_klawisz(unsigned char klawisz, int x, int y) {
 DEBUG("nacisnieto KLAWISZ");
 // naciśnięcie q lub ESCAPE (kod ASCII)
 // klawisze specjalne zdefiniowane w GL/glut.h (np. GLUT_KEY_DOWN)
 switch (klawisz) {
  case 'q':
  case 0x1b:
   // niszczymy okno GLUT'owe
   glutDestroyWindow(okno);
   // kończymy program
   exit(0);
  case 'F':
   // przejście w tryb pełnoekranowy
   glutFullScreen();
   // w zasadzie przedtem powinniśmy zapamiętać:
   // glutGet(GLUT_WINDOW_WIDTH) glutGet(GLUT_WINDOW_HEIGHT)
   // oraz położenie okna aby muc wrócić do trybu okienkowego przy pomocy:
   // glutReshapeWindow() i glutPositionWindow()
   break;
  case 'w':
   kamera.z += step_xyz;
   kamera.is_new = 1;
   break;
  case 's':
   kamera.z -= step_xyz;
   kamera.is_new = 1;
   break;
  case 'a':
   kamera.x += step_xyz;
   kamera.is_new = 1;
   break;
  case 'd':
   kamera.x -= step_xyz;
   kamera.is_new = 1;
   break;
  case 'f':
   kamera.y += step_xyz;
   kamera.is_new = 1;
   break;
  case 'r':
   kamera.y -= step_xyz;
   kamera.is_new = 1;
   break;
  case 'i':
   printf ("kamera = {%f,%f,%f,   %f,%f,   %f};\n",
    kamera.x, kamera.y, kamera.z,
    kamera.obr_xz, kamera.obr_yz, kamera.zoom);
   kamera.is_new = 1;
   break;
 }
}

// funkcja wywoływana po naciśnięciu przycisku myszy
int x_0=0;
int y_0=0;
void nacisnieto_mysz(int przycisk, int stan, int x, int y) {
 DEBUG("nacisnieto MYSZ");
 if (przycisk == 3) { // kółko w górę
  if (kamera.zoom > max_zoom) {
```

```
   kamera.zoom -= step_zoom;
   kamera.is_new = 1;
  }
 } else if (przycisk == 4) { // kółko w dół
  if (kamera.zoom < min_zoom) {
   kamera.zoom += step_zoom;
   kamera.is_new = 1;
  }
 } else {
  x_0=x;
  y_0=y;
 }
 // UWAGA: wartości przycisków odpowiadających ruchowi kółka
 //         nie znalazłem w dokumentacji a odczytałem doświadczalnie
 //          - mogą się różnić pomiędzy implementacjami
}

// funkcja wywoływana przy poruszeniu myszki z wciśniętym przyciskiem
void przesunieto_mysz(int x,int y) {
 kamera.obr_xz -= (x-x_0)*step_obr;
 kamera.obr_yz -= (y-y_0)*step_obr;
 kamera.is_new = 1;
 x_0=x;
 y_0=y;
}

/// UWAGA: aby nie zaciemniać kodu tak rozbudowane sterowanie jak powyżej
///         jest tylko dla GLUTa, oczywiście nie stoi nic na przeszkodzie
///         aby dopisać je poprzez analogię do kodu dla innych bibliotek

// typ wyliczeniowy dla obsługi menu
enum {
 UKRYJ_PRZOD,
 UKRYJ_TYL,
 UKRYJ_NIC,
 RYSUJ_PUNKTY,
 RYSUJ_LINIE,
 RYSUJ_WYPELNIENIA,
 PRZELACZ_SWIATLA,
 PRZELACZ_SWIATLO_0,
 PRZELACZ_SWIATLO_1,
 PRZELACZ_SWIATLO_2,
 MGLA,
 PRZELACZ_TEXTURA_0,
 PRZELACZ_TEXTURA_1,
};
// funkcja wywoływana po wybraniu pozycji menu
void akcja_menu (int opcja) {
 switch (opcja) {
  case UKRYJ_TYL:
   glEnable(GL_CULL_FACE);
   glCullFace(GL_BACK);
   break;
  case UKRYJ_PRZOD:
   glEnable(GL_CULL_FACE);
   glCullFace(GL_FRONT);
   break;
  case UKRYJ_NIC:
   glDisable(GL_CULL_FACE);
   break;
```

```
  case RYSUJ_WYPELNIENIA:
   glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
   break;
  case RYSUJ_LINIE:
   glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
   break;
  case RYSUJ_PUNKTY:
   glPolygonMode(GL_FRONT_AND_BACK, GL_POINT);
   break;
  case PRZELACZ_SWIATLA:
   if (glIsEnabled(GL_LIGHTING))
    glDisable(GL_LIGHTING);
   else
    glEnable(GL_LIGHTING);
   break;
  case PRZELACZ_SWIATLO_0:
   if (glIsEnabled(GL_LIGHT0))
    glDisable(GL_LIGHT0);
   else
    glEnable(GL_LIGHT0);
   break;
  case PRZELACZ_SWIATLO_1:
   if (glIsEnabled(GL_LIGHT1))
    glDisable(GL_LIGHT1);
   else
    glEnable(GL_LIGHT1);
   break;
  case PRZELACZ_SWIATLO_2:
   if (glIsEnabled(GL_LIGHT2))
    glDisable(GL_LIGHT2);
   else
    glEnable(GL_LIGHT2);
   break;
  case MGLA:
   if (glIsEnabled(GL_FOG))
    glDisable(GL_FOG);
   else
    glEnable(GL_FOG);
   break;
  case PRZELACZ_TEXTURA_0:
   glActiveTexture(GL_TEXTURE0);
   if (glIsEnabled(GL_TEXTURE_2D))
    glDisable(GL_TEXTURE_2D);
   else
    glEnable(GL_TEXTURE_2D);
   break;
  case PRZELACZ_TEXTURA_1:
   glActiveTexture(GL_TEXTURE1);
   if (glIsEnabled(GL_TEXTURE_2D))
    glDisable(GL_TEXTURE_2D);
   else
    glEnable(GL_TEXTURE_2D);
   break;
 }
}

/// funkcja wywoływana przez timer
void print_fps(int arg) {
 int fps = licznik_klatek - arg;
 if (fps<0)fps +=1000; // gdy licznik się przekręcił
```

```
 printf("FPS: %d\n", fps);

// za 1000ms wywołaj ponownie przekazując jako argument
 // obecny stan licznika
 glutTimerFunc (1000, &print_fps, licznik_klatek);

// warto byc takze swiadomym mozliwosci uzycia glutGet(GLUT_ELAPSED_TIME);
}


// funkcja główna
int main (int argc, char **argv) {
 // inicjalizacja GLUTa i przekazanie do niego argumentów wywołania
 glutInit(&argc, argv);
 // ustawienie trybu wyświetlania
 glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE | GLUT_ALPHA | GLUT_DEPTH);
 // ustawienie początkowego rozmiaru okna GLUTowego
 glutInitWindowSize(500, 300);
 // ustawienie położenia okna GLUTowego (lewy górny róg)
 glutInitWindowPosition(0, 0);
 // utworzenie okna
 okno = glutCreateWindow("tytul okna");

// przejście w tryb pełnoekranowy
 //glutFullScreen();

// określenie funkcji odpowiedzialnej za rysowanie sceny
 glutDisplayFunc(&narysuj_scene);
 // określenie funkcji wykonywanej gdy nie ma zdarzeń do obsłużenia
 glutIdleFunc(&narysuj_scene);
 // okręslenie funkcji wykonywane przy zmianie rozmiaru okna
 glutReshapeFunc(&ustaw_rozmiar_sceny);

// określenie funkcji wywoływanej po naciśnięciu przycisku na klawiaturze
 glutKeyboardFunc(&nacisnieto_klawisz);
 // określenie funkcji wywoływanej po naciśnięciu przycisku myszy
 glutMouseFunc(&nacisnieto_mysz);
 // określenie funkcji wywoływanej przy poruszeniu myszki z wciśniętym przyciskiem
 glutMotionFunc(&przesunieto_mysz);
 // gdy chemy przechwytywac ruch bez naciskania przysisku to glutPassiveMotionFunc()

// tworzymy pierwsze submenu
 int submenu1 = glutCreateMenu(akcja_menu);
 glutAddMenuEntry("przod", UKRYJ_PRZOD);
 glutAddMenuEntry("tyl", UKRYJ_TYL);
 glutAddMenuEntry("nic", UKRYJ_NIC);
 // tworzymy kolejne submenu
 int submenu2 = glutCreateMenu(akcja_menu);
 glutAddMenuEntry("punkty", RYSUJ_PUNKTY);
 glutAddMenuEntry("linie", RYSUJ_LINIE);
 glutAddMenuEntry("wypelnienia", RYSUJ_WYPELNIENIA);
 // tworzymy kolejne submenu
 int submenu3 = glutCreateMenu(akcja_menu);
 glutAddMenuEntry("all", PRZELACZ_SWIATLA);
 glutAddMenuEntry("0", PRZELACZ_SWIATLO_0);
 glutAddMenuEntry("1", PRZELACZ_SWIATLO_1);
 glutAddMenuEntry("2", PRZELACZ_SWIATLO_2);
 // tworzymy kolejne submenu
 int submenu4 = glutCreateMenu(akcja_menu);
 glutAddMenuEntry("0", PRZELACZ_TEKSTURA_0);
```

```
 glutAddMenuEntry("1", PRZELACZ_TEXTURA_1);
 // tworzymy menu glowne
 glutCreateMenu(akcja_menu);
 glutAddSubMenu("Ukryj: ", submenu1);
 glutAddSubMenu("Rysuj jako: ", submenu2);
 glutAddSubMenu("Przelacz swiatlo: ", submenu3);
 glutAddSubMenu("Przelacz jedn. teksturujaca: ", submenu4);
 glutAddMenuEntry("Mgla", MGLA);

// menu przypięte do prawego klawisza myszy
 glutAttachMenu (GLUT_RIGHT_BUTTON);

// po 1000ms wywolujemy funkcję print_fps
 glutTimerFunc(1000, &print_fps, 0);

// inicjalizacja OpenGL
 inicjalizuj_GL();

// przygotowanie obiektów dla funkcji rysującej
 przygotuj_scene();

// uruchomienie pętli przetwarzającej zdarzenia GLUT
 glutMainLoop();
}
```

OPENGL-INIT.C

```
/**

plik odpowiedzialny za obsługę "czystego" OpenGL - ustawienia

**/

/// jeżeli chemy skorzystać z shader'ów to włączamy plik z funkcjami obsługującymi ich ↩
ładowanie
#ifdef USE_SHADER
#include "opengl-load_shader.c"
#endif

/// nagłówki podstawowe OpenGL
// OpenGL (-lGL)
#include <GL/gl.h>
// GL Utility library (-lGLU)
#include <GL/glu.h>

/// funkcja inicjalizująca OpenGL _po_ utworzeniu jego okna
void inicjalizuj_GL() {
 // czyszczenie tła kolorem RGBA
 // kolor lekko czerwony, lekko przezroczysty
 // w GLUT (niekiedy) tło do przezroczystości z tego co za oknem, w SDL czarne
 glClearColor(0.0f, 0.4f, 0.0f, 0.3f);
 // czyszczenie bufora głębi zadaną wartością
 glClearDepth(1.0);

// włączamy porównywanie głębi
 // funkcją tą możemy włączać wiele innych funcjonalności OpenGL (do wyłączania ↩
 glDisable)
 glEnable(GL_DEPTH_TEST);
```

```
  // ustawienie funkcji porównującej głębie
  glDepthFunc(GL_LESS);

 // ustawiamy tryb shadingu na "smooth color", czyli:
  // kolor liczony dla każdego wierzchołka i interplolowany na resztę wielokąta
  glShadeModel(GL_SMOOTH);

 // ustawienia blendingu, czyli mówimy jak chemy używac kanału alfa
  glBlendFunc( GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA );
  glEnable( GL_BLEND );

 // włączamy ukryweanie niewidocznych powierzchni
  glEnable(GL_CULL_FACE);
  // ukrywamy tył
  glCullFace(GL_BACK);
  // strona przednia to ta której wierzchołki rysowane są zgodnie z ruchem wskazówek ⤸
 zegara
  glFrontFace(GL_CW);

 // włączamy efekty oświetlenia
  //glEnable( GL_LIGHTING );

 // ustawiamy mgłę - kolor
  float kolor_mgly[] = {0.3f, 0.35f, 0.3f, 0.0f}; // lekko zielonkawa
  glFogfv( GL_FOG_COLOR, kolor_mgly );
  // aby uzyskać mgłę pozaobiektami warto na ten sam kolor ustawić glClearColor()
  // oraz typ (wykładnicza) i jej gęstość; w przypadku mgły liniowej
  // zamiast gęstości ustawiamy Z początku i końca mgły
  glFogi( GL_FOG_MODE, GL_EXP );
  glFogf( GL_FOG_DENSITY, 0.25f );
  // ale jej nie włączamy
  glDisable( GL_FOG );

 // ustawiamy redenrowanie wypełniające dla obu stron wielokątów
  glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);

 // ustawienia specyficzne dla implementacji
  // konkretnie ustawiamy najbardziej poprawne korygowanie kolorów i tekstór przy liczeniu ⤸
 perspektywy
  glHint( GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST );

 // wymuszamy wykonanie komend OpenGL w tym momencie
  glFlush();
}


/// struktura globalna przechowująca informacje o położeniu kamery
struct parametry_kamery {
  float x, y, z; // punkt z którego patrzymy
  float obr_xz; // obrót w plaszczyźnie podłogi (przeciwnie do wskazówek zegara)
  float obr_yz; // pochylanie góra-dół (dodatnie: patrzymy do góry)
  // ogólnie możliwy jest także obrót kamery wokół własnej osi (z)
  float zoom; // kąt widzenia (w y, bo w x zadany przez ratio okna)
  char is_new;
};
/// wartości domyślne (początkowe) dla kamery
//static struct parametry_kamery kamera = {0,0,0, 0,0, 40.0f};
static struct parametry_kamery kamera = {-1.999999,0.000000,-6.090007,   -38.939995,-2.999999,⤸
40.000000};
```

```
/// funcja ustawiająca kamere
void ustaw_kamere() {
  // uzyskujemy informację o wysokości i szerokości view portu
  GLfloat view_port_info[4];
  glGetFloatv(GL_VIEWPORT, view_port_info);

// ustawiamy jako bierzącą macierz rzutowania (gdzie i z kąd patrzy kamera)
  glMatrixMode(GL_PROJECTION);
  // ustawiamy _ją_ na identyczność
  glLoadIdentity();

// ustawiamy perspektywę rzutowania:
  //    * kąt widzenia w Y
  //    * proporcje ekranu
  //    * Z przedniej płaszczyzny ocinania
  //    * Z tylniej płaszczyzny obcinania
  gluPerspective(kamera.zoom, view_port_info[2]/view_port_info[3], 0.1f, 100.0f);
  // warto wspomnieć o gluOrtho2D() która sprawia że wyświetlane na ekranie
  // są obiekty o współrzędnych x,y odpowiadajęce podanemu do niej prostokątowi

// ustawiamy położenie kamery (w DX jest to osobna macierz - VIEW)
  // przesunięcie w prawo, w górę i w głąb ekranu
  glTranslatef(kamera.x, kamera.y, kamera.z);

// obrót kamery w płaszyźnie yz (względem osi x)
  glRotatef(kamera.obr_yz, 1, 0, 0);
  // obrót kamery w płaszyźnie xz (względem osi y)
  glRotatef(kamera.obr_xz, 0, 1, 0);

// oczywiście oprócz wyżej zaprezentowanej metody działań na macierzach
  // możemy operować jawnie na nich przy reprezentacji w postaci tablic jednowymiarowych
  // (wpisujemy ciórkiem kolejne wiersze), do załadowania przygotowanej takiej macierzy
  // do OpenGL (np jako GL_PROJECTION) służy funckja glLoadMatrixf() lub glMultMatrix()
  // wykonana po odpowiednim glMatrixMode

kamera.is_new = 0;

// przełączamy bierzącą macierz na macierz świata (domyślna wartość)
  glMatrixMode(GL_MODELVIEW);
}


/// funcja ustawiająca rozmiar sceny
void ustaw_rozmiar_sceny(int szerokosc, int wysokosc) {
  // ustawiamy rozmiar widoku: lewy gorny róg = (0,0) oraz wysokość i szerokość
  // w ogólności może on być mniejszy od rozmiaru okna
  glViewport(0, 0, szerokosc, wysokosc);

// ustawiamy położenie i inne parametry kamery
  // (musimy przeliczyć bo zależą od rozmiaru view point'a)
  ustaw_kamere();
}
```

OPENGL-DRAW-VERTEX.C

```
/**

plik odpowiedzialny za obsługę "czystego" OpenGL - rysowanie
```

```
prezentuje techniki poświęcone buforom vertexów i texturowaniu

WARTO JESZCZE ZAPOZNAĆ SIĘ Z:
 * obiekty buforowe vertexów i pikseli czyli korzystamy z RAMu karty graficznej

**/

/// nagłówek biblioreki libglpng odpowiedzialnej za
/// obsługę plików png dla OpenGL
#include <GL/glpng.h>

/// zmienna globalna przechowująca licznik klatek
int licznik_klatek = 0;

/// zmienne globalne trzymające informacje o teksturach i jej plikach
GLuint tekstura1, tekstura2;
pngInfo tekstura_info;


// typ struktury opisującej typ "podzbioru" informacji o vertexie
// ma wyłącznie charakter pomocniczy do zapisu deklaracji vdesc
typedef struct {
 GLint size;       // liczba składowych "pozdbioru"
 GLenum type;      // typ danych składowych "podzbioru"
 GLsizei offset;   // przesunięcie w bajtach od początku werteksu
} vertex_subset_desc;

/// struktura opisująca typ naszego vertexa
// swoją drogą warto też zauważyć sposób deklaracji struktury "const"
const struct {
 const vertex_subset_desc punkt;
 const vertex_subset_desc tekstura1;
 const vertex_subset_desc tekstura2;
} vdesc = {
 {3, GL_FLOAT, 0},
 {2, GL_FLOAT, 3*sizeof(GLfloat)},
 {2, GL_FLOAT, (3+2)*sizeof(GLfloat)}
};

/// typ struktury reprezentującej punkt naszego vertexu
typedef struct {
 GLfloat punkt[3];
 GLfloat tekstura1[2];
 GLfloat tekstura2[2];
} vertex_point;

vertex_point kwadrat [4] = {
 {{ 0.0f,  4.0f, 0.0f},   {0.0f, 0.0f},   {0.0f, 0.0f}},
 {{ 4.0f,  4.0f, 0.0f},   {1.0f, 0.0f},   {2.0f, 0.0f}},
 {{ 4.0f,  0.0f, 0.0f},   {1.0f, 1.0f},   {2.0f, 2.0f}},
 {{ 0.0f,  0.0f, 0.0f},   {0.0f, 1.0f},   {0.0f, 2.0f}},
};

// trywialna tablica indeksów - tylko po to aby pokazać jak działa
GLbyte kwadrat_indeksy [4] = {0, 1, 2, 3};

// w zasadzie możnaby (a chyba nawet warto) reprezentować taki obiekt jako klasę która ↩
oprócz
// tablicy elementów vertex_point, struktury vdesc i tablicy indeksów
// zawierałaby także informację o typie tekstury (np. GL_TEXTURE_2D), o sposobie rysowania
```

```
// vertexu (np. GL_QUADS), opcje teksturowania (powielanie, ramka itp)


/// funkcja przygotowująca scenę
void przygotuj_scene(void) {
 /// wczytanie pliku tekstury
 tekstura1 = pngBind("tekstura1.png", PNG_NOMIPMAP, PNG_ALPHA, &tekstura_info, GL_CLAMP, ⤸
 GL_NEAREST, GL_NEAREST);
 tekstura2 = pngBind("tekstura2.png", PNG_NOMIPMAP, PNG_ALPHA, &tekstura_info, GL_CLAMP, ⤸
 GL_NEAREST, GL_NEAREST);

/// załaduj i aktywuj tablice położeń wierzchołków
 glVertexPointer(
  vdesc.punkt.size,     // ilość (kolejnych) elementu tablicy położeń
  vdesc.punkt.type,     // typ elementu tablicy położeń
  sizeof(vertex_point), // okresowość występowania początku tab. poł.
                        // w tablicy opisującej cały vertex
  (void*)kwadrat +  vdesc.punkt.offset // adres pierwszego elementu tab. poł.
 );
 glEnableClientState(GL_VERTEX_ARRAY);
 glEnable(GL_TEXTURE_2D);

/// załaduj i aktywuj tablice współrzędnych tekstury
 // dla pierwszej jednostki teksturującej
 glActiveTexture(GL_TEXTURE0); // przełączamy aktywną jednostkę teksturującą
 glClientActiveTexture(GL_TEXTURE0); // i po stronie klienta
 glTexCoordPointer(
  vdesc.tekstura1.size,     // ilość (kolejnych) elementu tablicy wsp. teks.
  vdesc.tekstura1.type,     // typ elementu tablicy wsp. teks.
  sizeof(vertex_point),      // okresowość występowania początku tab. wsp. teks.
                            // w tablicy opisującej cały vertex
  (void*)kwadrat +  vdesc.tekstura1.offset
                           // adres pierwszego elementu tab. wsp. teks.
 );
 glEnableClientState(GL_TEXTURE_COORD_ARRAY); // aktywujemy tablicę współrzędnych
 glEnable(GL_TEXTURE_2D); // właczamy teksturowanie 2d
 glBindTexture(GL_TEXTURE_2D, tekstura1); // ustawiamy teksture

// dla drugiej jednostki teksturującej
 glActiveTexture(GL_TEXTURE1); // przełączamy aktywną jednostkę teksturującą
 glClientActiveTexture(GL_TEXTURE1); // i po stronie klienta
 glTexCoordPointer(
  vdesc.tekstura2.size,     // ilość (kolejnych) elementu tablicy wsp. teks.
  vdesc.tekstura2.type,     // typ elementu tablicy wsp. teks.
  sizeof(vertex_point),      // okresowość występowania początku tab. wsp. teks.
                            // w tablicy opisującej cały vertex
  (void*)kwadrat +  vdesc.tekstura2.offset
                           // adres pierwszego elementu tab. wsp. teks.
 );
 glEnableClientState(GL_TEXTURE_COORD_ARRAY); // aktywujemy tablicę współrzędnych
 glEnable(GL_TEXTURE_2D); // właczamy teksturowanie 2d
 glBindTexture(GL_TEXTURE_2D, tekstura2); // ustawiamy teksture

// wykorzystanie kilku jednostek teksturujących umożliwia korzystanie z prawdziwego ⤸
 wieloteksturowania
 // (zamiast renderowania kilkakrotnie tej samej sceny z przeźroczystością i zmienianą ⤸
 teksturą)
 // gdybyśmy nie korzystali z vertexów definiowanych tablicami wpółrzędne dla innych ⤸
 niż pierwsza
 // jednostek teksturujących wpółrzędne tekstury ustawiane byłyby funkcjami z grupy ⤸
```

```
 glMultiTexCoord()

// tak jak pokazano w opengl-main-glut.c możemy wyłączyć używanie danej jednostki ⤸
 teksturującej
  // (np. dla danego obiektu) np. poprzez deaktywację przypisanego do niej typu tekstury:
  // glActiveTexture(GL_TEXTURE1); glDisable(GL_TEXTURE_2D);

// można także ładować całą grupę przy pomocy glInterleavedArrays
  // jednak jest to rozwiązanie znacznie mniej elastyczne gdyż wymaga wyboru
  // jednego z narzuconych formatów wertexa
};

/// funkcja rysująca (przerysowująca) scenę
void narysuj_scene() {
  // jeżeli nastąpiła zmiana ustawień kamery to ją wdrażamy
  if (kamera.is_new) ustaw_kamere();

// czyścimy ekran (ustawionym wcześniej) kolorem oraz czyścimy bufor głębokości
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

// ustawiamy bierzącą macierz (GL_MODELVIEW) na identyczność
  glLoadIdentity();

// mnożymy bierzącą macierz przez macierz przesunięcia (x, y, z)
  glTranslatef(-1.5f,0.0f,-6.0f);

// wyświetlamy 4 kolejne elementy tablicy vertexów począwszy od elementu 0
  glDrawArrays(GL_QUADS, 0, 4);
  // jest również funkcja umożliwiająca rysowanie kilku obiektów zapisanych
  // w jednej tablicy - glMultiDrawArrays przyjmeje ona tablice określające
  // położenia pierwszego vertexu kolejnych obiektów i jego długości

glTranslatef(-1.0f,0.0f,1.0f);
  glScalef(0.3f, 0.3f, 0.3f);

// możemy też oczywiście korzystać z indeksowanych tablic werteksów
  // wprowadzenie tablicy indeksów służy redukcji rozmiaru tablic
  // werteksów poprzez wyeliminowanie powtarzających się wierzchołków

// rysujemy trójkąt w oparciu o 3 ostatnie punkty kwadratu
  glDrawElements (GL_TRIANGLE_FAN, 3, GL_UNSIGNED_BYTE, kwadrat_indeksy + 1);
  // warto też zwrócić uwagę na glMultiDrawElements() i glDrawRangeElements()

// przy korzystaniu z tablic verteksów i związanych z nimi ustawień warto
  // wspomnieć o funkcjach glPushClientAttrib()/glPopClientAttrib()
  // umożliwiających odkładanie na stos (i przywracanie) tychże ustawień
  // podobnie jak glPushMatrix() czyni z macierzami świata / widoku

// licznik FPS
  licznik_klatek++;
  if (licznik_klatek == 1000) {
   licznik_klatek = 0;
#if MODE == SDL
   static czas_ostatniego_wypisania = 0;
   GLint czas1 = SDL_GetTicks();
   GLint czas2 = czas1 - czas_ostatniego_wypisania;
   czas_ostatniego_wypisania = czas1;
   printf("FPS %g\n", 1000000.0 / czas2); // 1000 ms/s * 1000 frame / x s
#endif
  }
```

```
// wywołujemy (zależną od wyżejpoziomowego mechanizmu) funkcję rysującre scenę do ⤸
 okna
#ifdef DRAW_TO_SCREEN
 DRAW_TO_SCREEN;
#elif MODE == GLUT
 glutSwapBuffers();
#elif MODE == SDL
 SDL_GL_SwapBuffers();
#elif MODE == GLX
 glXSwapBuffers(oknoGLX.wyswietlacz, oknoGLX.okno);
#else
# error "Couldn't determine DRAW_TO_SCREEN function"
#endif
}
```

OPENGL-DRAW-SIMPLE.C

```
/**

plik odpowiedzialny za obsługę ”czystego” OpenGL - rysowanie
ukazane są proste metody rysowania oraz korzystanie z list

ponadto pokazuje jak można wykorzystać GLSL
 (w oparciu o ”opengl-load_shader.c” includowany w ”opengl-init.c”)
 aby w tym przykładzie użyć shaderów zamiast stałego potoku
 do opcji kompilacji należy dodać: ”-lGLEW -D USE_SHADER”

**/

/// nagłowki ogólne
#include <math.h>

/// nagłówek biblioreki libglpng odpowiedzialnej za
/// obsługę plików png dla OpenGL
#include <GL/glpng.h>

/// zmienne globalne trzymające informacje o teksturach i jej plikach
GLuint tekstura;
pngInfo tekstura_info;

/// zmienna globalna przechowująca licznik klatek
int licznik_klatek = 0;

/// funkcja przygotowująca scenę
#define STORZEK 11 // coś >0
void przygotuj_scene(void) {
 /// przygotowujemy obiekt storzka
 // aby w każdej ramce nie musieć przeliczać storzka
 // (funkcje trygonometryczne nie są szybkie)
 // przygotowujemy go wcześniej i traktujemy jako gotowy obiekt
 glNewList(STORZEK, GL_COMPILE);
 // listy możemy także usuwać - glDeleteLists()
 // możemy też uzyskiwać zbiór zadanej ilości identyfikatorów - glGenLists
 // oraz sprawdzać czy identyfikator jest wolny - glIsList
 // rysować będziemy kółko złożone z trójkątów
 glBegin( GL_TRIANGLE_FAN );
  // środek (wspólny wierzchołek wszystkich trójkątów)
```

```
   glColor3f(1.0f,1.0f,0.0f); glVertex3f( 0.0f, 0.0f, 2.0f);
   // punkty na okręgu (pozostałe wierzchołki trójkątów)
   GLint i;
   for (i=0; i<20; i++) {
    glColor3f(sin(i*0.0785f),cos(i*0.0785f),cos(1.57f-i*0.0785f));
    glVertex3f( sin(i*0.157f), cos(i*0.157f), 0.0f);
   }
   for (i=20; i<=41; i++) {
    glColor3f(1-cos(i*0.0785f),1-sin(i*0.0785f),sin(1.57f-i*0.0785f));
    glVertex3f( sin(i*0.157f), cos(i*0.157f), 0.0f);
   }
  glEnd( );
  glEndList();

/// wczytanie pliku tekstury
 tekstura = pngBind("tekstura.png", PNG_BUILDMIPMAPS, PNG_ALPHA, &tekstura_info, GL_CLAMP, ⤸
 GL_LINEAR, GL_LINEAR);
 // UWAGA: nie wiem dlaczego ale nie wszystkie PNG wczytują się poprawnie

// korzystamy tutaj z funkcji przypisującej od rzu teksturę do obiektu teksturowego OpenGL,
 // normalnie obiekt taki jest tworzony poprzez wywołanie glGenTextures
 // obiekty te ustawiane jako aktywne (przełączane) są poprzez wywołanie funkcji
 // glBindTexture(), oprócz samej tekstury przechowują one też informacje ustawiane
 // funkcjami glTexParameter... takie jak powtarzalność itp cechy

// bez zastosowania tego mechanizmu tekstury wraz z wszystkimi ich własnościami
 // muszą być ustawiane dla kolejnych obiektów niezależnie poprzez wywołanie
 // funkcji typu glTexImage2D i glTexParameter, aby wczytać teksturę do użycia
 // w taki sposób można skorzystać z funkcji pngLoad

// ustawiamy powtarzalność tekstury na boki
 glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
 // wypełnianie kolorem ramki w górę
 glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP);
 // ustalamy kolor ramki
 GLint kolor[] = {255,255,255,255};
 glTexParameteriv(GL_TEXTURE_2D, GL_TEXTURE_BORDER_COLOR, kolor);
 // przy pomocy glTexParameteri możemy też ustawić filtrowanie tekstur (np. mipmapping)
 // ustawaiamy trójliniowe wygładzanie (liniowa interpolacja najbliższych mipmap
 // a następnie filtrowanie liniowe) dla filtra pomniejszającego
 glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR_MIPMAP_LINEAR);
 // i liniowe powiększanie dla powiększającego
 glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
 // warto zaznaczyć że niektóre powyższe ustawienia mogą być dokonane
 // od razu w funkcji pngBind

// mechanizm obiektów teksturowych pozwala także na ustawienie priorytetu tekstury
 // wpływającego na przechowywanie jej w pamięci karty graficznej glAreTexturesResident()
 // (normalnie wybór tekstur przechowywanych w pamięci karty opiera się na
 // mechanizmach typu LRU lub MFU znanych z mechanizmów swappowania stron pamięci ⤸
 operacyjnej)

// warto także wspomnieć o ficzerach nie omówionych tutaj szczegółowo, takich jak:
 // kompresja tekstur, tekstury kubiczne (teksturyujące sześcian), teksturach 3d
 // (pozwalających na łatwiejsze teksturowanie obiektów z nierównościami gdyż
 // zawierających kolejne warstwy tekstóry dobierane w zażności jak głeboko
 // położony element obiektu teksturujemy), tekstur głębi

#ifdef USE_SHADER
 // jeżeli używamy shaderów to należy przygotować i ustawić odpowiedni program dla GPU ⤸
```

```
 ..
 GLenum typy[] = {GL_VERTEX_SHADER, GL_FRAGMENT_SHADER};
 const char* pliki[] = {"shader.vert", "shader.frag"};
 glewInit();
 MakeShadersProgram( typy, pliki, 2, 1 );
 #endif
};

/// funkcja rysująca (przerysowująca) scenę
void narysuj_scene() {
 // jeżeli nastąpiła zmiana ustawień kamery to ją wdrażamy
 if (kamera.is_new) ustaw_kamere();

// czyścimy ekran (ustawionym wcześniej) kolorem oraz czyścimy bufor głębokości
 glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

// ustawiamy bierzącą macierz (GL_MODELVIEW) na identyczność
 glLoadIdentity();
 /** sprawdzenie że to rzeczywiście GL_MODELVIEW
 GLint tmp;
 glGetIntegerv (GL_MATRIX_MODE, &tmp);
 printf("%d from %d %d %d\n", tmp, GL_MODELVIEW, GL_PROJECTION, GL_TEXTURE); **/

// mnożymy bierzącą macierz przez macierz przesunięcia (x, y, z)
 glTranslatef(-1.5f,0.0f,-6.0f);
 // możemy także skorzystać z mnożenia przez macierze:
 //    * skalowania glScalef(x, y, z);
 //    * obrotów glRotatef(kat w stopniach, x , y, z); zadany punkt i (0,0,0) określają oś
 obrotu

// zaczynamy rysować łamaną zamkniętą bez wypełnienia
 // poprzez wskazywanie kolejnych jej punktów wraz z ich kolorami
 // punkty określamy za pomocą współrzędnych:
 //    x (rosnie w prawo), y (rośnie w górę), z (rośnie w przód)
 glBegin(GL_LINE_LOOP);
  glLineWidth(1.0f);
  // zobacz też glLineStipple
  glColor3f(1.0f,0.0f,0.0f); glVertex3f( 0.0f, 0.0f, 0.0f);
  glColor3f(1.0f,1.0f,0.0f); glVertex3f( 1.0f, 0.0f, 0.0f);
  glColor3f(0.0f,1.0f,0.0f); glVertex3f( 1.0f, 1.0f, 0.0f);
  glColor3f(0.0f,1.0f,1.0f); glVertex3f(-1.0f, 0.0f, 0.0f);
  glColor3f(0.0f,0.0f,1.0f); glVertex3f( 0.0f,-1.0f, 0.0f);
  glColor3f(1.0f,1.0f,1.0f); glVertex3f( 1.0f,-1.0f, 0.0f);
  // należy wspomnieć że zamiast bezpośredniego określania kolorów,
  // możemy je określić w buforze i posługiwać się tylko numerami
 glEnd(); // kończymy rysować

// kolejny obiekt jest o
 //    licznik_klatek/300.0f na prawo od poprzedniego
 //    1.0 do przodu od poprzedniego
 // OSTRZEŻENIE: prędkość animacji obiektu zależna od fps a nie czasu !!!
 //                     nie powinno się tak robić ze względu na różne
 //                     zachowanie w zależności od szybkości komputera
 glTranslatef(licznik_klatek/300.0f,0.0f,1.0f);
 // półprzezroczysty prostokąt (prawa ścianka wgłąb sceny) z nałożoną tekturą
 // patrz ustawienia blendingu w funkcji inicjalizującej
 glColor4f(0.5f,0.5f,1.0f,0.5f);
 // jako teksturę 2D ustawiamy wczytaną wcześniej teksturę
 // poprzez wywołania tej funkcji możemy różne obiekty teksturować innymi obrazkami
 glBindTexture(GL_TEXTURE_2D, tekstura);
```

```
 // tekstura będzie modulowana kolorem obiektu (jest to ustawienie całego
 // środowiska teksturującego a nie ustawienie przypisane konkretnej teksturze)
 glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);
 // warto bliżej przyjrzeć się możliwym ustawieniom gdyż pozwalają
 // na uzyskanie wielu ciekawych efektów
 // włączamy teksturowanie
 glEnable(GL_TEXTURE_2D);
 glBegin(GL_QUADS);
  glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f, 1.0f, 0.4f);
  glTexCoord2f(2.5f, 0.0f); glVertex3f( 1.0f, 1.0f, 0.0f);
  glTexCoord2f(2.5f, 1.5f); glVertex3f( 1.0f, 0.0f, 0.0f);
  glTexCoord2f(0.0f, 1.5f); glVertex3f(-1.0f, 0.0f, 0.4f);
 glEnd();
 // wyłaczamy teksturowanie (inne obiekty mają być tylko kolorowane)
 glDisable(GL_TEXTURE_2D);

// warto wspomnieć o uzyskiwaniu przezroczystości obiektu z wykorzystaniem
 // kanału alfa tekstury, efekt ten możemy uzyskać na dwa sposoby:
 // * poprzez zastosowanie standardowego blendingu (należy uważać na kolejność
 //      rysowania obiektów - obiekty najbliżej kamery muszą być rysowane na końcu)
 // * poprzez testowanie wartości alfa fragmentów (IMHO trochę gorszy efekt -
 //      bardziej ostre przejścia) przy pomocy funkcji glEnable (GL_ALPHA_TEST); i
 //      glAlphaFunc(tryb_porownania, wartosc);

// tym razem pozycjonujemy znowu od początku a nie od ostatniego obiektu
 glLoadIdentity();
 // zamiast tego przed kazda modyfikacja mozemy dawac glPushMatrix();
 // a po narysowaniu tak przesuniętych obiektów glPopMatrix();
 glTranslatef(0.5f,0.0f,-7.0f+licznik_klatek/140.0f);
 glCallList(STORZEK);

// licznik FPS
 licznik_klatek++;
 if (licznik_klatek == 1000) {
  licznik_klatek = 0;
#if MODE == SDL
  static czas_ostatniego_wypisania = 0;
  GLint czas1 = SDL_GetTicks();
  GLint czas2 = czas1 - czas_ostatniego_wypisania;
  czas_ostatniego_wypisania = czas1;
  printf("FPS %g\n", 1000000.0 / czas2); // 1000 ms/s * 1000 frame / x s
#endif
 }

// wywołujemy (zależną od wyżejpoziomowego mechanizmu) funkcję rysującre scenę do ⤸
 okna
#ifdef DRAW_TO_SCREEN
 DRAW_TO_SCREEN;
#elif MODE == GLUT
 glutSwapBuffers();
#elif MODE == SDL
 SDL_GL_SwapBuffers();
#elif MODE == GLX
 glXSwapBuffers(oknoGLX.wyswietlacz, oknoGLX.okno);
#else
# error "Couldn't determine DRAW_TO_SCREEN function"
#endif
}
```

OPENGL-DRAW-LIGHT.C

```
/**

plik odpowiedzialny za obsługę "czystego" OpenGL - rysowanie
w tym przykładzie ukazane jest korzystanie z świateł

**/


// warto zwrócić uwagę na możliwość wykorzystania tekstur zawierających
// informację o jasnosnoći) do uzyskiwania efektów oświetleniowych,
// metoda ta pozwala w łatwy sposób ominąć ograniczenie systemu oświetlenia
// polegające na obliczaniu światła tylko dla wierzhołków, jednak nie pozwala
// na dynamiczne oświetlenie sceny

/// nagłowki ogólne
#include <math.h>
#include <string.h>

/// zmienna globalna przechowująca licznik klatek
int licznik_klatek = 0;

// funkcja wypisująca napis (tylko GLUT)
void umiesc_napis(char *tekst, GLfloat x, GLfloat y) {
#ifdef GLUT_INIT
 glColor3f(1.0f, 0.0f, 1.0f);
 glRasterPos3f(x, y, -5.0f);

int i, dlugosc = strlen(tekst);
 for (i = 0; i < dlugosc; i++)
  glutBitmapCharacter(GLUT_BITMAP_9_BY_15, tekst[i]);
#endif
}

/// funkcja przygotowująca scenę
#define STOZEK 11 // coś >0

void przygotuj_scene(void) {
 // przygotowujemy obiekt storzka
 // aby w każdej ramce nie musieć przeliczać storzka
 // (funkcje trygonometryczne nie są szybkie)
 // przygotowujemy go wcześniej i traktujemy jako gotowy obiekt
 glNewList(STOZEK, GL_COMPILE);
 // rysować będziemy kółko złożone z trójkątów
 glBegin( GL_TRIANGLE_FAN );
  // środek (wspólny wierzchołek wszystkich trójkątów)
  glColor3f(1.0f,1.0f,0.0f); glVertex3f( 0.0f, 0.0f, 2.0f);
  // punkty na okręgu (pozostałe wierzchołki trójkątów)
  GLint i;
  for (i=0; i<20; i++) {
   glColor3f(sin(i*0.0785f),cos(i*0.0785f),cos(1.57f-i*0.0785f));
   glVertex3f( sin(i*0.157f), cos(i*0.157f), 0.0f);
  }
  for (i=20; i<=41; i++) {
   glColor3f(1-cos(i*0.0785f),1-sin(i*0.0785f),sin(1.57f-i*0.0785f));
   glVertex3f( sin(i*0.157f), cos(i*0.157f), 0.0f);
  }
 glEnd( );
 glEndList();
```

```
/// ustawienia świateł
 // reflektor (Spot Light)
 float kolor[] = {0.0f, 1.0f, 0.5f, 1.0f};
 glLightfv(GL_LIGHT0, GL_DIFFUSE, kolor);
 float pozycja[] = {0.0f, 0.0f, 4.0f, 1.0f};
 // ostatnia wartość =1.0f oznacza umieszczenie źródła na scenie
 glLightfv(GL_LIGHT0, GL_POSITION, pozycja);
 float kierunek[] = {0.0f, 0.0f, -1.0f};
 glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, kierunek);
 // kąt świecenia reflektora
 glLightf(GL_LIGHT0, GL_SPOT_CUTOFF, 5.0f);
 // skupienie światła
 glLightf(GL_LIGHT0, GL_SPOT_EXPONENT, 10.0f);
 // liniowe zanikanie światła wraz z odelgłością
 glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, 0.01f);

// światło kierunkowe o promieniach równoległych
 // czyli źródło w nieskończoności (światło słoneczne)
 // w odróżnieniu od pozostałych daje światło dla odbłysków (GL_SPECULAR)
 kolor[0]=1.0; kolor[1]=0.0; kolor[2]=0.6;
 glLightfv(GL_LIGHT1, GL_SPECULAR, kolor);
 pozycja[3]=0;
 glLightfv(GL_LIGHT1, GL_POSITION, pozycja);
 glLightfv(GL_LIGHT1, GL_SPOT_DIRECTION, kierunek);

// światło punktowe (żarównka na scenie)
 kolor[0]=0.0; kolor[1]=0.0; kolor[2]=0.6;
 glLightfv(GL_LIGHT0 + 2, GL_DIFFUSE, kolor);
 pozycja[2]=-3.0f; pozycja[3]=1;
 glLightfv(GL_LIGHT0 + 2, GL_POSITION, pozycja);
 // kwadratowe zanikanie światła wraz z odelgłością
 glLightf(GL_LIGHT0 + 2, GL_QUADRATIC_ATTENUATION, 0.3f);
 // warto zauważyć inny sposób podawania numeru światła ...
 // jest on przydatny zwłaszcza gdy nasza implementacja obsługuje więcej niż 8 źródeł
 // skoro mówimy o tym ograniczeniu liczby źródeł światła to warto wspomieć że
 // po narysowaniu kawałka sceny możemy je przeunąć i pozmieniać aby narysować inny)

// oświetlenie ogóle (AMBIENT)
 kolor[0]=0.5; kolor[1]=0.4; kolor[2]=0.0;
 glLightModelfv(GL_LIGHT_MODEL_AMBIENT, kolor);
 // można jest stosować też przy normalnych światłach,
 // ale nie ma to większego zastosowania gdyż nie ma kierunku z którego ono pada

// funkcje z grupy glLightModel mogą posłużyć także do aktywacji drugorzędnego
 // koloru odbicia (secondary specular color), ponadto poprzez glEnable(GL_COLOR_SUM)
 // i funkcje z rodziny glSecondaryColor możemy uzyskać kontolę nad ustawieniem
 // drugorzędnego koloru wierzchołka (dla tablic wertexów -   glSecondaryColorPointer)

/// aktywacja świateł
 glEnable(GL_LIGHTING);
 glEnable(GL_LIGHT0);
 glEnable(GL_LIGHT0 + 2);
};

/// funkcja rysująca (przerysowująca) scenę
void narysuj_scene() {
 // jeżeli nastąpiła zmiana ustawień kamery to ją wdrażamy
 if (kamera.is_new) ustaw_kamere();
```

```
// czyścimy ekran (ustawionym wcześniej) kolorem oraz czyścimy bufor głębokości
 glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

glLoadIdentity();

// napisy informacyjne
 char info[100];
 sprintf(info, "GL_LIGHTING: %d", glIsEnabled(GL_LIGHTING));
 umiesc_napis(info, -14.0f, -3.3f);
 sprintf(info, "GL_LIGHT0: %d", glIsEnabled(GL_LIGHT0));
 umiesc_napis(info, -14.0f, -4.0f);
 sprintf(info, "GL_LIGHT1: %d", glIsEnabled(GL_LIGHT1));
 umiesc_napis(info, -14.0f, -4.7f);
 sprintf(info, "GL_LIGHT2: %d", glIsEnabled(GL_LIGHT2));
 umiesc_napis(info, -14.0f, -5.4);
 // bardziej porządną implementację warstwy napisowej można znaleźć w ”Podstawa silnika ↩
 3D”

float mat[] = {0.9f, 0.9f, 0.9f, 1.0f};
 float mat_zero[] = {0.0f, 0.0f, 0.0f, 1.0f};

// przelatujący stożek
 glTranslatef(0.0f, 0.0f, -7.0f+licznik_klatek/140.0f);
 glMaterialfv(GL_FRONT, GL_DIFFUSE, mat);
#ifdef GLUT_INIT
 glutSolidTeapot(1.0f);
#else
 glCallList(STOZEK);
#endif
 mat[0]=1.0f; mat[1]=1.0f; mat[2]=1.0f;
 //glMaterialfv(GL_FRONT, GL_DIFFUSE, mat);

glLoadIdentity();
 glScalef(0.5, 0.5, 0.5);
 // dodatkowe storzki demonstrujące różne ustawienia materiałów,
 // czyli informacji o tym jak obiekt odbija światło

// stożek oświetlony sam z siebie
 // kolorami takimi jak zdefiniowane (śledzenie koloru)
 glTranslatef(0.0f, 3.0f, -5.0f);
 glEnable(GL_COLOR_MATERIAL);
 glColorMaterial(GL_FRONT, GL_EMISSION);
 glCallList(STOZEK);
 glDisable(GL_COLOR_MATERIAL);
 glMaterialfv(GL_FRONT, GL_EMISSION, mat_zero);
 // zamiast śledzenia koloru morzna włączyć GL_EMISSION poprzez
 // glMaterialfv z wskazanym kolorem świecenia

// stożek na wysoki połysk
 mat[0]=0.0f; mat[1]=0.0f; mat[2]=0.8f;
 glTranslatef(0.0f, -2.5f, 0.0f);
 glMateriali(GL_FRONT, GL_SHININESS, 7);
 glMaterialfv(GL_FRONT, GL_SPECULAR, mat);
 glCallList(STOZEK);
 glMaterialfv(GL_FRONT, GL_SPECULAR, mat_zero);
 glMateriali(GL_FRONT, GL_SHININESS, 0);

// stożek na wysoki oświetlany światłem rozproszonym
 mat[0]=1.0f; mat[1]=1.0f; mat[2]=1.0f; mat[3]=0.0f;
 glMaterialfv(GL_FRONT, GL_AMBIENT, mat);
```

```
 glTranslatef(0.0f, -2.5f, 0.0f);
 glCallList(STOZEK);
 glMaterialfv(GL_FRONT, GL_AMBIENT, mat_zero);

// prawdziwy efekt materiału (np. złota) uzyskuje się poprzez
 // ustawienie odpowiednich wartości wszystkich parametrów:
 // * GL_AMBIENT (współczynniki odbicia światła otaczającego
 //      - padającego tak samo z wszystkich kierunków)
 // * GL_DIFFUSE (współczynniki odbicia światła rozproszonego
 //      - padającego z określonych kierunków, odbijanego równomiernie)
 // * GL_SPECULAR (współczynniki odbicia światła odbłysków
 //      - padającego z określonych kierunków, odbijanego kierunkowo)
 // * GL_SHININESS (stopnia połysku
 //      - wielkości plamy powstałej z światła odbłysków)


// licznik FPS
 licznik_klatek++;
 if (licznik_klatek == 1000) {
  licznik_klatek = 0;
 }

// wywołujemy (zależną od wyżejpoziomowego mechanizmu) funkcję rysującre scenę do ⤸
 okna
#ifdef DRAW_TO_SCREEN
 DRAW_TO_SCREEN;
#elif MODE == GLUT
 glutSwapBuffers();
#elif MODE == SDL
 SDL_GL_SwapBuffers();
#elif MODE == GLX
 glXSwapBuffers(oknoGLX.wyswietlacz, oknoGLX.okno);
#else
# error "Couldn't determine DRAW_TO_SCREEN function"
#endif
}
```

OPENGL-LOAD_SHADER.C

```
/** plik odpowiedzialny za obsługę ładowania i kompilacji shaderów
 *
 * shadery są programami działającymi na karcie graficznej i zmieniającymi
 * standardowy sposób przetwarzania grafiki
 *
 * ogólnie wyróżniamy dwa typy shaderów:
 *    - vertex shader, który operuje na wierzchołkach
 *    - fragment shader, który operuje na już zrasteryzowanych punktach
 *      (pixelach mających 3 wymiarowe współrzędne i kolor ...)
 *
 * w przypadku OpenGL język w którym pisane są te programy nazywa się GLSL
 * i od strony składniowej przypomina C, jednak z zupełnie inną biblioteką
 * (nie mamy funkcji biblioteki standardowej, za to są specyficzne funkcje tego języka)
 *
 * pełna specyfikacja tego języka dostępna jest na http://www.opengl.⤸
 org/documentation/glsl/
**/

#include <GL/glew.h>
```

```c
#include <GL/glut.h>

#include <stdio.h>
#include <string.h>
#include <errno.h>

// for fstat()
#include <sys/types.h>
#include <sys/stat.h>
// for close()
#include <unistd.h>
// for open()
#include <fcntl.h>
// for mmap() munmap()
#include <sys/mman.h>

#ifndef _POSIX_MAPPED_FILES
#error Wymagany system z obsługą POSIXowego mapowania plików do pamięci - funkcja mmap
#endif

/// funkcja mapująca plik tekstowy do pamięci,
/// zwraca wskaźnik na początek pliku,
/// jeżeli length != NULL ustawia w nim długość pliku
char* MapTextFile(const char *file_path, size_t *length, char null_end) {
 // otwieranie pliku - dostęp przez deskryptor
 int desc = open(file_path, O_RDONLY);
 if (desc == -1) {
  return NULL;
 }

// uzyskiwanie informacji o pliku
 // robimy to w mniej przenośny ale sprytniejszy sposób niż przy uzyciu fopen, fseek, ↩
 ftell
 struct stat info;
 fstat(desc, &info);

// wczytanie pliku, a dokładniej zamapowanie go do pamięci
 // to równierz jest mniej przenośny, ale sprytniejszy sposób niż malloc, fread
 char* addr;
 if (null_end) {
  info.st_size++;
  addr = (char*)mmap(NULL, info.st_size, PROT_READ|PROT_WRITE, MAP_PRIVATE, desc, 0);
 } else {
  addr = (char*)mmap(NULL, info.st_size, PROT_READ, MAP_SHARED, desc, 0);
 }
 if (addr == MAP_FAILED) {
  close(desc);
  return NULL;
 }
 // ustawienie kończącego NULLa
 if (null_end) {
  addr[info.st_size]='\0';
 }

// zamknięcie deskryptora
 close(desc);

// przekazanie informacji o długości alokowanego obszaru
 if (length != NULL) {
  *length= (size_t) info.st_size;
```

```
 }

// zwrócenie adresu
 return addr;
}

/// funkcja tworzy shader o podanym typie, na podstawie podanego pliku
/// zwracając jego numeryczny identyfikator
GLuint MakeShader(GLenum type, const char *file_path) {
 const GLchar *txt;
 size_t txt_len;
 GLuint shader;

// wczytanie tekstu shadera
 txt = MapTextFile(file_path, &txt_len, 1);
 if (txt == NULL) {
  fprintf(stderr, "%s:%d - Nie można wczytać pliku shadera %d (%d: \"%s\")\n",
   __FILE__, __LINE__, file_path, errno, strerror(errno));
  return 0;
 }

// utworzenie shadera
 shader = glCreateShader(type);
 // ustawienie źródła shadera
 glShaderSource(shader, 1, &txt, NULL);
 // kompilacja shadera
 glCompileShader(shader);

// odmapowanie pliku z tekstem shadera
 munmap((void*)txt, txt_len);

return shader;
}

/// funkcja tworzy "program" na podstawie zadanych tablic z typami shaderów i ścieżkami do ⤸
plików je definiujących
/// funkcja ponaddto instaluje utworzony program (jeżeli use !=0)
/// funkcja zwraca numeryczny identyfikator zlinkowanego programu
GLuint MakeShadersProgram(const GLenum types [], const char * files [], int num, char use) {
 // types -> tablica wartości typu GL_VERTEX_SHADER, GL_FRAGMENT_SHADER, ...
 // files -> tablica napisów będących ściezkami do shaderów
 // przed wywołaniem tej funkcji należy robić glewInit(); (przynjamniej gdy używamy ⤸
 "Mesa")

// tworzy program
 GLuint program = glCreateProgram();

// podłączamy do niego shadery
 int i;
 for (i=0; i<num; i++) {
  glAttachShader( program, MakeShader(types[i], files[i]) );
 }

// linkujemy i aktywujemy program
 glLinkProgram(program);
 if(use) glUseProgram(program);

return program;
}
```

Podstawa silnika 3D

Jest to praca którą wykonywałem jako pracę zaliczeniową na "Programowanie Gier Komputerowych". Głównym zagadnieniem rozwiązanym w projekcie był culling na podstawie rzutowanego na płaszczyznę stożka widzenia, wraz z dynamicznym ładowaniem obiektów. Projekt nie ma statusu ukończonego (spora część zaplanowanej funkcjonalności nie jest zaimplementowana), jednak najprawdopodobniej nie będzie rozwijany dalej. Projekt realizowany był z wykorzystaniem OpenGL.

Pobierz[1028]

Bliżej sprzętu i jądra

- test_libusb.c - program demonstrujący komunikację poprzez USB z wykorzystaniem biblioteki libusb
- ups_off.c - program sprawdzający stan zasilania z ups'u (poprzez port COM) i wykonujący wyłączenie systemu gdy jest zasilanie z zrodla awaryjnego ...
- zawieszacz.c - co zrobić aby system zajmował się tylko naszym programem ...
- priorytety.c - ustalanie priorytetów (nice i priorytet czasu rzeczywistego)
- czekanie.c - jak wykonywać dość precyzyjne opóźnienia w programie ...
- modul.c, Makefile - demonstracyjny moduł jądra Linux wraz z Makefile umożliwiającym jego zbudowanie ... pokazuje inicjacje i usuwanie modułu, korzystanie z urządzeń znakowych oraz przerwań; więcej o modułach: The Linux Kernel Module Programming Guide[1029], Linux Device Drivers[1030]
- zabawy_pamiecia.c - plik obrazuje sposób alokacji pamięci przez system Linux
- ftdi_ds18b20.tar.gz[1031] - zestaw modułów dla Linuxa składających się na sterownik do obsługi termometru ds18b20 opartego na interfejsie OneWire z wykorzystaniem układu FTDI FT245BM do emulacji portu OneWire (rozpowszechniane na zasadach GNU GPL)
- vkbd.tar.gz[1032] - moduł wirtualnej klawiatury, umożliwiającej odbieranie znaków i key-kodów wpisywanych do urządzenia w /dev tak jakby przychodziły z podsystemu input od rzeczywistej lokalnej klawiatury (rozpowszechniane na GNU GPL)

Oczywiście oprócz zaprezentowanych w powyższych przykładach technik programowania modułów jądra. Linux oferuje wiele innych udogodnień - np. kolejki kfifo (bufory cykliczne). Należy zaznaczyć też iż programowanie jądra wymaga szczególnej uwagi gdyż wiele z mechanizmów chroniących programy uruchamiane w przestrzeni użytkownika tutaj nie funkcjonuje. Ponadto szczególną uwagę należy zwrócić na niemożność używania wielu funkcji w obsłudze przerwań.

test_libusb.c

```
// gcc test_libusb.c -lusb

#include <stdio.h>
#include <usb.h>

int main(void)
{
 /// struktury dla magistali i urządzenia
 struct usb_bus *bus;
 struct usb_device *dev;

/// inicjalizacja
 usb_init();
 usb_find_busses();
 usb_find_devices();

/// wyszukiwanie urządzenia
 char find = 0;
 for (bus = usb_busses; bus && find == 0; bus = bus->next)
  for (dev = bus->devices; dev; dev = dev->next)
   if ( dev->descriptor.idVendor == 0x0403
     && dev->descriptor.idProduct == 0x6001
```

```
//          && dev->descriptor.iSerialNumber == 0x0000
       )
    {
     find = 1;
     break;
    }
  if (find != 1) {
   puts("nie znaleziono urządzenia");
   return -1;
  }
  free(bus);  // bus nie będzie nam już potrzebne, znalezione urządzenie mamy w dev

/// obsługa urządzenia
  usb_dev_handle *usbdev;
  usbdev = usb_open(dev);
  if (!usbdev) {
   puts("błąd otwarcia urządzenia");
   return -2;
  }

// tworzymy interfejs na urządzeniu
  if (usb_claim_interface(usbdev, 0)) {
   puts("błąd utworzenia interfejsu");
   puts(" upewnij się że nik nie korzysta z urządzenia, spóbój `rmmod ftdi_sio`");
   usb_close (usbdev);
   return -3;
  }

/// komunikacja z urządzeniem
  // ustawienia konfiguracyjne:
  // usb_control_msg(usbdev, requesttype, request, value, index, *bytes, size, timeout);
  // wysłanie danych:
  // usb_bulk_write(usbdev, ep, *bytes, size, timeout);
  // odczyt danych:
  // usb_bulk_read(usbdev, ep, *bytes, size, timeout);
  // UWAGA: bufor odbiorczy musi być zdeklarowany jako static unsigned char []

// reset device
  if (usb_control_msg(usbdev, 0x40, 0, 0, 0, NULL, 0, 3000) != 0)
   puts("błąd resetu układu");
// if (usb_control_msg(usbdev, 0x40, 3, 16696, 0, NULL, 0, 3000) != 0)
//    puts("Błąd ustawienia prędkości");

// send data
  char * aaa="abbcb";
  int i;
  for (i=0; i<5; i++) {
   printf("Wysłano: %d\n", usb_bulk_write(usbdev, 0x02, aaa+i, 1, 3000));
   // UWAGA: wysyłana porcja danych nie może być zbyt duża !!!
   sleep(1);
  }

puts("Czekam na dane wejściowe");
  int j, k;
  for (j=0; j<5; j++) {
   // read data
   static unsigned char buf[3];
   i = usb_bulk_read(usbdev, 0x81, buf, 3, 3000);
   // UWAGA: bufor mniejszy od 3 znaków nie ma sensu ze względu na sposób transmisji ...
   if (i==3) {
```

```
    printf("%d : ", j);
    for (k=0; k<i; k++)
     printf("%X ", buf[k]);
    puts("");
   } else {
    j--;
   }

  usleep(200000);
 }

/// zakończenie
 usb_release_interface(usbdev, 0);
 usb_close (usbdev);
 return 0;
}
```

UPS_OFF.C

```
// program przeznaczony do wywolywaniqa z skryptu reboot (na runlevel 6) ...
// program sprawdza stan zasilania ups'a Fideltronic Ares 1000
// jezeli system dziala na zasilaniu z UPS (praca awaryjna)
// nastapi wylaczenie ups'a oraz wykonanie polecenia halt,
// w przeciwnym razie program zakonczy sie ... pozwalajac na dokonanie reboot'u

// gdy program wywolany nie jest w runlevelu 6 poinformuje o stanie zasilania i zakonczy ⤸
dzialanie

#include <stdio.h>
#include <sys/io.h>
#include <unistd.h>

#define RS_PORT 0x3f8
int main(int argv, char *argc[]) {
 iopl (3);

// ustawiamy DTR na stan wysoki
 outb(1, RS_PORT + 4);

// odczytujemy CTS - informacja o stanie zasilania
 if ( inb(RS_PORT + 6) & (1 <<4) ) {
  puts("Normalny stan zasilania - nic nie robie");
  return 0;
 } else {
  puts("Awaryjny stan zasilania - czekam");
  // jezeli runlvel !=6 nie kontynuje ...
  if ( 0 != system("test `runlevel | cut -f2 -d' '` = 6")) {
   puts("Runlevel != 6 => koncze");
   return -2;
  } else {
   sleep(5);
   // ponownie sprawdzam stan zasilania
   if ( ! ( inb(RS_PORT + 6) & (1 <<4) ) ) {
    puts("Nadal awaryjny stan zasilania => wylaczam");

    // ustawiamy stan wysoki na RTS - wylaczenie UPS za 5s
    outb(2, RS_PORT + 4);
```

```
      // wykonanie halt
      system("halt -d -f -i");
      return -3;
    } else {
      puts("Powrot do normalnego stanu zasilania ...");
      return -1;
    }
  }
 }
}
```

ZAWIESZACZ.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metode wylaczania przerwań - przydaje się to gdy na k ótko potrzebujemy
 * działania w mocnym trybie czasu rzeczywistego (jest to rozwiązanie dużo mocniejsze
 * niż priorytet -20, gdzie program nawet o dużo niższym (np. dodatnim) priorytecie
 * ma prawo dorwać się do głosu, tu nie może tego zrobić nawet jądro
 * - wszystkie przerwania poza niemaskowalnymi są zablokowane
 *
 */

#include <sys/io.h> // w tym siedzi funkcja iopl();
#include <stdio.h>

int main (int argv, char *args[]) {

int a=999999999;

FILE *plik;
 plik = fopen("tekst.txt","w");

iopl (3); // musimy przejac wladze nad portami ;-),
          // inaczej mamy blad ochrony pamieci i nasz wspanialy program z komenda "cli"
          // nie bedzie sie uruchamial, ale do tego to potrzeba nam root'a ...

asm("cli"); // - wylacza przerwania


while (1) {
  // nistety jezeli chcielibysmy cos czytac lub pisac to przerwania beda wlaczone :(((
  //scanf("%d",&a);
  //printf("%d", a);
  //fprintf(plik, "%d", a);

  // a to taki bezpiecznik, ktory zapewni ze kom sie odiwesi ;-)
  a--;
  if (a == 0) break;
 }

asm("sti"); // - wlacza przerwania

return 0;
}
```

PRIORYTETY.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania w języku C / C++
 * prezentuje ustalanie nice oraz priorytetów czasu rzeczywistego
 *
 * program z oczywistych wzgledów powinien być uruchamiany z prawami root'a ...
 *
 */


#include <sched.h>
#include <sys/resource.h>
#include <sys/types.h>
#include <errno.h>

int main ()
{
  // nice = -20
  // ta funkcja działa tylko na bierzącym procesie - nie pobiera PIDu
  // tak naprawde wywolywanie jeje w tym programie nie ma sensu
  // gdyz zaraz zmienimy polityke kolejkowania (i ststyczny priorytet) na niebslugujaca nice'a
  setpriority (PRIO_PROCESS, 0, -20);



/// priorytet statyczny (rt_priority), priorytet czasu rzeczywistego

// deklaracje zmiennych
  pid_t pid;
  struct sched_param sp;


pid = getpid(); // pobranie identyfikatora biezacego procesu
  sp.sched_priority = sched_get_priority_max(SCHED_FIFO);
  // uwaga: przy wysokim priorytecie (tak jak w tym przykladzie)
  //               moze nie byc mozliwe zabicie programu
  // przy niszych prirorytetach proces powinno dac sie zabic kombinacja Ctrl+C ...

// ustawienie trybu szeregowania oraz priorytetu
  // funkcja ta moze byc wykorzystana do nadawania priorytetu dowolnemu procesowi
  // (nie tylko biezacemu), wystarczy ze znamy jego PID
  if ( sched_setscheduler(pid, SCHED_RR, &sp) ) {
   perror("ERROR");
   return -1;
  }
  // w tym miejscu warto wpomnieć o metodach szeregowania:
  // * SCHED_RR jest to proste szeregowanie z kwantem czasu (proces dostaje czas,
  //      a po jego wykorzystaniu lub oddaniu procesora brany jest kolejny
  //      a ten idzie na koniec kolejki)
  // * SCHED_FIFO podobnie do SCHED_RR tyle że jedynym powodem dla którego proces może
  //      przestać być wykonywany jest to iż sam zrezygnuje (jego kwant czasu jest ⤸
 nieskonczony)
  // * SCHED_OTHER zależy od systemu, najczęsciej podobnie do SCHED_RR tyle że miejsce
  //      na którym ląduje w kolejce zalezy od tego jak dużo używał CPU (im więcej tym ⤸
 dalej)
  // * SCHED_SPORADIC (nie implementowana aktualnie w Linuxie) jest polityką szeregowania
  //      typu serwera sporadycznego (procesy mogą chodzić z wysokim priorytetem przez zadany
  //      kwant czasu potem - do czasu odnowienia priorytetu serwera sporadycznego ich
  //      priorytet jest obniżany)
```

```
  //
  // warto zauważyć iż SCHED_FIFO wraz z sched_yield() pozwala na implementowanie ⤸
 szeregowania
  // powodującego przełączanie zadań tylko gdy ione tego sobie rzyczą (podobnie jak w ⤸
 PicOSie ...)
  //
  // oczywiście dla każdego z priorytetów istnieje osobna kolejka i dopóki są gotowe ⤸
 zadania
  // w kolejce o wyższym priorytecie, kolejka o niższym nie ma nic do powuiedzenia


// metoda ta wydaje sie rownie skuteczna jak opisane w zawieszacz.c wylaczanie przerwan,
  // przy czym jest rozwiazaniem elegantrzym
  // oraz umozliwiajacym korzystanie z wielu funkcji ktore uniewaznialy tamta metode


// zastapienieaktualnie wykonywanego programu przez program ktory chcemy tak uruchomic ...
  execlp("ps", "ps", "-l", "-A", 0);

// wiecej o szertegowaniu procesow pod Linuxem:
  // http://www.localnet.pl/man/man2/sched_setscheduler.2.html
}
```

CZEKANIE.C

```
/*
 * plik ma na celu prezentację ciekawych zagadnień z programowania w języku C / C++
 * prezentuje metody wykonywania precyzyjnych przerw w programach
 *
 * uwaga: aby uzyskać naprawde dokładne przerwy należy prezentowane tu metody wyczekiwania
 * łączyć z metodami powodującymi uprzywilejowanie naszego procesu w dostępie do ⤸
 procesora
 * opisanymi w zawieszacz.c i priorytety.c
 *
 */

#include <stdio.h>


// funkcja zwracajaca wartosc rejestru zliczajacego takty procesora
unsigned long long czas() {
   asm( "rdtsc" );
}

struct timeval tv_1;
unsigned long long czas2() {
  gettimeofday ( &tv_1, NULL );
  return (unsigned long long)( tv_1.tv_sec * 1000000 + tv_1.tv_usec );
    // zmiesci sie to nam gdyz 10^6 * 2^32 < 2^64
}


int main (int argv, char *args[]) {

int ile = atoi(argv[1]); // czas czekania w us

// czekanie aktywne w oparciu o licznik taktów i pomiar czasu czekania ...
  // czekanie to moze dzialac nawet z precyzja kilku nanosekund
```

```
    // - zalezy to od predkosci taktowania procesora ...
    struct timeval tv_1, tv_2;
    gettimeofday ( &tv_1, NULL );
     tmp1 = czas() + 3000*ile;
     while ( czas() < tmp1);
    gettimeofday ( &tv_2, NULL );
    printf("CZAS [us]: %li\n",
     (tv_2.tv_sec * 1000000 + tv_2.tv_usec) - (tv_1.tv_sec * 1000000 + tv_1.tv_usec) );

  // jak widzimy pomiar ten jest dosc dokladny (jest sytematyczne zawyzenie
    // ale wiaze sie ono z czasem wykonywania dodatkowych instrukcji ...
    // dla czasów oczekiwań > 4us można zatem zrealizować z wykorzystaniem gettimeofday
    // nalezy wtedy korzystac z czas2() oraz nie podawac mnoznika 3000
    // ktory jest zwiazany z predkoscia procesora (w przykladzie 3GHz
    // mnoznik ten mozna obliczas z wykorzystaniem polaczenia obu metod

  // wiecej na ten temat w zrodlach "Sterowanie magnetowidem"

  // na koniec trzeba tez wspomniec o nanosleep(), ktora zgodnie z dokumentacja man powinna
    // dla czasow <2ms i polityce szeregowania czasu rzeczywistego uzywac aktywnego czekania
    // jednak z przeprowadzonych testow wynika ze przynajmniej nie zawsze tak sie dzieje ...

  return 0;
  }
```

MODUL.C

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z programowania modułów jądra ⤸
 systemu Linux (w C)
 * prezentuje wypisywanie komunikatów dmesg (printk), umieszczenie informacji o module
 *    (pokazywanych przez modinfo), obsługę urządzeń znakowych, wykorzystywanie portów ⤸
 oraz przerwań
 *
 * urządzenie znakowe wykorzystywane przez ten moduł nalezy utworzyć poleceniem 'mknod ⤸
 plik c _DEV_ID_ 0'
 * (_DEV_ID_ zastępujemy oczywiście odpowiednim numerem)
 *
 * problemy z zaladownaiem - moze jakis modul (np. parport) uzywa tego portu ...
 * uwaga: wazne jest aby moduly parport nie byly ladowane wcale
 * (po ich usunieciu nie dzialaja instrukcje pisania do portu LPT)
 *
 * aby załadować moduł wydajemy jako root polecenie insmod sciezka/do/naszego/modulu
 * uwaga: nalezy liczyć się z ryzykiem że załadowanie błędnego modułu może wywalić ⤸
 cały system
 *
 *
 * więcej o tworzeniu modułów: http://www.faqs.org/docs/kernel/ http://lwn.⤸
 net/Kernel/LDD3/
 *
 * program zainspirowany zajęciami z Programowania 2 ( 2003/2004) na Wydziale Fizyki UW:
 * http://www.fuw.edu.pl/~pablo/s/
 * pomyślany o współpracy z płytką wykorzystywaną na tych zajęciach (umożliwia m.in. ⤸
 zmianę stanu ACK)
 *
 */
```

```
// mknod plik c _DEV_ID_
// mknod urzadzenie c 242 0
// echo -e "Ala ma kota\000" > urzadzenie

// problemy z zaladownaiem - moze jakis modul (np. parport) uzywa tego portu ...
// uwaga: wazne jest aby moduly parport nie byly ladowane wcale
// (po ich usunieciu nie dzialaja instrukcje pisania do portu LPT)
// dlatego nalezy dodac je do blacklisty w /etc/modprobe.d/blacklist:
//     blacklist parport
//     blacklist parport_pc

// przerwanie wywoływane jest poprzez zbocze narastajace na ACK (pin 10) - rozwarcie ACK od ⤸
masy

#include <linux/module.h>
#include <linux/fs.h>
#include <linux/ioport.h>
#include <asm/io.h>
#include <asm/irq.h>
#include <linux/signal.h>
#include <linux/sched.h>
#include <linux/interrupt.h>
#include <asm/uaccess.h>

// informacje o naszym module
MODULE_LICENSE("BSD");
MODULE_DESCRIPTION("Modul demonstracyjny ...");
MODULE_AUTHOR("Robert Paciorek");

// informacje konfiguracyjne
#define _MOD_NAME_ "wyswietlacz"
#define _DEV_ID_ 242
#define _LPT_BASE_ 0x378
// standardowy port to 0x378, ale modol paraport_pc przesuwa obsluge portu LPT0 na 0x778 (⤸
tryb ECR portu typu ECP)
#define _LPT_LEN_ 3
#define _LPT_IRQ_ 7


// zmienna zawierająca informacje o tym jak daleko zaszło uruchamianie modułu (przydatne ⤸
przy jego usuwaniu);
static int poziom_uruchomienia = 0;
static char sa_dane = 0;

// funkcja wykonywana przy usuwaniu modułu - uswuwanie modułu (definicja ciała - patrz ⤸
niżej)
void cleanup_module( void );

// funkcja której użyjemy do obsługi przerwania (definicja ciała - patrz niżej)
static irqreturn_t obsluz_przerwanie( int irq_no, void* dev_id, struct pt_regs* regs );

// funkcja wywoływana gdy następuje zapis do urządzenia znakowego    (definicja ciała - ⤸
patrz niżej)
static ssize_t char_dev_write( struct file *file, const char __user *buf, size_t count, ⤸
loff_t *ppos );

// funkcja wywoływana gdy następuje odczyt z urządzenia znakowego    (definicja ciała - ⤸
patrz niżej)
static ssize_t char_dev_read( struct file *file, char __user *buf, size_t count, loff_t *ppos ⤸
```

```
);

// funkcja wywoływana gdy następuje otwarcie urządzenia znakowego    (definicja ciała - ⤸
patrz niżej)
static int char_dev_open (struct inode *inode, struct file *file);

// struktura ustawien pliku urzadzenia znakowego
static struct file_operations char_dev_conf = {
 .owner  = THIS_MODULE,
 .read  = char_dev_read, // tutaj określamy funkcję wywoływaną przy czytaniu
 .write  = char_dev_write, // tutaj określamy funkcję wywoływaną przy zapisie
 .open  = char_dev_open, // ... przy otwarciu
 // lseek read    write readdir poll ioctl mmap open flush release fsync
};

// funkcja wykonywana przy ładowaniu modułu - inicjalizacja modułu
int init_module( void ) {
 printk(KERN_INFO "Ladowanie modulu rozpoczete\n");
 // zmienna przechowująca kody powrotu funkcji ...
 int kontrolna;

// sprawdzamy dostępność i dokonujemy zarezerwowania interesującego nas zakresu portów
 kontrolna = request_region( _LPT_BASE_, _LPT_LEN_, _MOD_NAME_ );
 if ( !kontrolna ) {
  // był niedostępny ... kończymy
  printk(KERN_ALERT "Niemogę zarezerwować zakresu portów\n");
  cleanup_module();
  return -ENODEV;
 }
 poziom_uruchomienia++; // 1 zapamiętujemy postępy ...

// sprawdzamy dostępność przerwania, jeżeli dostępne to dokonujemy rezerwacji i ⤸
 wyznaczmy funkcję obsługi
 kontrolna = request_irq( _LPT_IRQ_, obsluz_przerwanie, 0, _MOD_NAME_, 0 );
 if ( kontrolna ) {
  // było niedostępne ... kończymy
  printk(KERN_ALERT "Niemogę zarezerwować przerwania\n");
  cleanup_module();
  return kontrolna;
 }
 poziom_uruchomienia++; // 2 zapamiętujemy postępy ...

// chcemy aby LPT (linia ACK) generowało przerwania
 outb( 0x0f, _LPT_BASE_ );
 outb( 0x1b, _LPT_BASE_ + 2 );

// sprawdzamy dostępność numeru urządzenia znakowego, jeżeli dostępne to dokonujemy ⤸
 rezerwacji
 kontrolna = register_chrdev( _DEV_ID_, _MOD_NAME_, &char_dev_conf );
 if ( kontrolna < 0 ) {
  // był niedostępny ... kończymy
  printk(KERN_ALERT "Niemogę zarezerwować urządzenia znakowego\n");
  cleanup_module();
  return kontrolna;
 }
 poziom_uruchomienia++; // 3 zapamiętujemy postępy ...

// konczymy inicjalizację modułu
 struct timespec now;
 getnstimeofday(&now);
```

```
printk(KERN_INFO "Ladowanie modulu zakonczone %d %d\n", now.tv_sec, now.tv_nsec);
 return 0;
}


static irqreturn_t obsluz_przerwanie( int irq_no, void* dev_id, struct pt_regs* regs ) {
 struct timespec now;
 getnstimeofday(&now);

printk(KERN_INFO "Otrzymałem przerwanie z LPT %d %d\n", now.tv_sec, now.tv_nsec);

return IRQ_HANDLED;
}

static ssize_t char_dev_write( struct file *file, const char __user *buf, size_t count, ⤸
loff_t *ppos ) {
 printk(KERN_INFO "wczytano: %s\n", buf);
 outb( buf[1], _LPT_BASE_ );
 return count;
}

static ssize_t char_dev_read( struct file *file, char __user  *buf, size_t count, loff_t ⤸
*ppos ) {
 if(sa_dane==1) {
  copy_to_user(buf, "Helo World\n", 11);
  // należy zauważyć że niemy bezpośredniego dostępu do danych w przestrzeni ⤸
  użytkownika,
  // ponadto ta funkcja kopiująca nie może być wykonywana w obsłudze przerwania
  // jest to spowodowane tym że dane z przestrzeni użytkownika mogą być np. wyswapowane
  sa_dane=0;
  return 11;
 } else {
  return 0;
 }
}

static int char_dev_open (struct inode *inode, struct file *file) {
 sa_dane = 1; // zaznaczamy ze mamy cos wypisac ...
 // gdyby nie to zabezpieczenie po zrobieniu cat nasze_urzadzenie pisaloby sie w ⤸
 nieskonczonosc
 return 0;
}

void cleanup_module( void ) {
 printk(KERN_INFO "Usuwanie modulu rozpoczete\n");
 switch (poziom_uruchomienia) {
  case 3:
   unregister_chrdev( _DEV_ID_, _MOD_NAME_ );
  case 2:
   outb( 0xb, _LPT_BASE_ + 2 ); // wyłącz generację przerwań z LPT
   free_irq( _LPT_IRQ_, 0 ); // zwolnij numer przerwania
  case 1:
   release_region( _LPT_BASE_, _LPT_LEN_ ); // zwolnij zakres portów
 }
 printk(KERN_INFO "Usuwanie modulu zakonczone\n");
}
```

Makefile

```
# plik Makefile służący do budowy modułów w katalogu poza drzewem jądra ...
# plik zainspirowany przykładem z "Linux Device Driver"

# wywolanie zwyczajne
ifeq ($(KERNELRELEASE),)

KERNELDIR=/lib/modules/$(shell uname -r)/build
PWD=$(shell pwd)

modules:
 make -C $(KERNELDIR) M=$(PWD) modules

clean:
 rm -rf *.o *.ko *.mod.c .tmp_versions *.cmd

# wywolanie przez Makefile z $(KERNELDIR)
else

obj-m := modul.o

endif
```

ZABAWY_PAMIECIA.C

```
/*
 * plik ma na celu zobrazowanie sposobu alokacji pamięci w systemie Linux
 *
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

const char * napis = "Ala ma kota";
int a;
unsigned int * wsk;
unsigned char * wsk2;


main(int argc, char *argv[], char *envp[]) {
 printf("adresy zmiennych przekazywane do main()\n");
 printf("\t%x %x %x\n", &argc, &argv, &envp);
 printf("zmienne przekazywane do main()\n");
 printf("\t%x %x %x\n", argc, argv, envp);
 // warto zauważyć że envp jest kontynuacją argv - nic ich nie odziela w pamięci ...
 printf("napisy na które wskazują\n");
 printf("\t%x %x %x\n", argv[0], envp[0], envp[1]);

printf("\n\n");


printf("fragment stosu związany z main()\n");
 for (wsk=&argc; wsk<argv[0]; wsk++)
  if (*wsk != 0)
   printf("%x : %x\n", wsk, *wsk);
 // tutaj zaczynają się tablice char
```

```
 for (wsk2=argv[0]; wsk2<argv[0]+20; wsk2++)
  printf("%x : %x (%c)\n", wsk2, *wsk2, *wsk2);

printf("\n\n");


int b;
 int c;
 printf("alokacja zmiennej globalnej a lokalnej\n");
 printf("\t%x %x %x\n", &a, &b, &c);
 // warto zwrocic uwage iż zmienna lokalna jest zaalokowana na stosie przed
 // tablicami będącymi argumentami main(), a zmienna globalna poza stosem
 // kolejne zmienne alokowane są coraz bradziej przed (stos rośnie w dół)
 // jeżeli którąś z tych zmiennych będzie tablica, a następnie przekroczymy jej zakres
 // będziemy zamazywać kolejne obszary pamięci w stronę wyższych adresów (argumenty ⤸
 main, ...)
 // jest to tak zwany błąd przepełnienia bufora - http://pl.wikipedia.⤸
 org/wiki/Przepełnienie_bufora

// warto także zwrócić uwagę iż taka metoda alokacji tablic narzuca ograniczenie na ich ⤸
 rozmiar
 // związane z ograniczeniem rozmiaru stosu, dlatego lepiej duże tablice alokować poza ⤸
 stosem
 char d[100];
 char e[atoi(argv[1])];

// gdy jako agrument podamy np. 20000000 (20MB)
 // program zakończy się naruszeniem ochrony pamięci
 printf("alokacje tablic\n");
 printf("\t%x %x %x\n", &c, d, e);

// natomiast alokacja mallociem ponad 20 razy większej tablicy jest nieszkodliwa
 char * tab = malloc(sizeof(*tab) * 400000001 );
 tab[1] = 20;
 tab[400000000]=30;
 printf("\t%x %x %d %d\n", &tab[1], &tab[400000000], tab[1], tab[400000000]);

// a na koniec napis - jest to w zasadzie tablica, ale jakiego rodzaju ?
 char *bo;
 bo = strstr (napis, "kota");

printf("DLUGOSC NAPISU poprzez wskazniki (niezupelnie napisu dla kodowan typu UTF8)\n");
 printf("\t%x %x %i\n", napis, bo, (int)((unsigned int)bo - (unsigned int)napis));
}
```

Metody numeryczne

Jest to zbiór funkcji prezentujących podstawy metod numerycznych. Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie zagwarantować że działają one w pełni poprawnie. Więcej o metodach numerycznych także na Strona o Metodach Numerycznych[1033].

- calkowanie.cc
- interpolacja.cc
- miejsca_zerowe.cc
- rownania_rozniczkowe_czastkowe.cc
- rownania_rozniczkowe_zwyczajne.cc
- rozniczkowanie.cc
- sortowanie.cc

— transformata_fouriera.cc
— uklady_rownan.cc
— wartosci_wlasne.cc

WARTOSCI_WLASNE.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody poszukiwania wartości własnych macierzy
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */


void pomnoz (double A[], double v[], double wynik[], int wymiar) {
// mnozy macierz [nxn] razy wektor [n]
 double suma;

for (int wiersz=0; wiersz<wymiar; wiersz++) {
  suma = 0;
  for (int kolumna=0; kolumna<wymiar; kolumna++) {
   suma += A[wiersz*wymiar + kolumna] * v[kolumna];
  }
  wynik[wiersz] = suma;
 }
}

int maxww(double A[], double v[], int n, double eps, int limit, double *x) {
 double v_new[n], norma_new, norma;
 int liczba_iteracji=1, i;

// rozklad (A - stala * Id) na gorno i dolno trojkatna – wykorzystac program przykladowy LU

while(true) {
  pomnoz (A, v, v_new, n);
  norma_new = 0;
  for (i=0; i<n; i++) {
   norma_new += v_new[i]*v_new[i];
  }

  norma = 0;
  for (i=0; i<n; i++) {
   v_new[i] = v_new[i] / sqrt(norma_new);
   norma += (v[i]-v_new[i])*(v[i]-v_new[i]);
  }
  if (norma > eps) {
   // sprawdzamy czy nie za dlugo
   if (liczba_iteracji++ > limit) return -1;
  } else {
   // mamy wynik !!!
   for (i=0; i<n; i++) {
    norma = 0; // norma to teraz suma
    for (int j=0; j<n; j++) {
```

```
     norma += v_new[j] * A[j*n + i];
    }
    v[i] = norma; // v to teraz wynik mnozenia v_new^T * A
   }
   *x=0;
   for (i=0; i<n; i++)
    *x += v[i]*v_new[i];
   norma=0;
   for (i=0; i<n; i++)
    norma += v_new[i]*v_new[i]; // norma to teraz dzielnik

   *x = *x / sqrt(norma);
   return liczba_iteracji;
  }

  // jezeli nadal kontynuujemy to przygotowywujemy nastepny krok
  norma = norma_new;
  for (i=0; i<n; i++)
   v[i] = v_new[i];
 }
}
```

UKLADY_ROWNAN.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody rozwiązywania układów równań typu A * x = b,
 * gdzie A - macierz n x n, x i b - wektory o długości n
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie ⤸
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */

// zamiana macierzy na macierz trojkatna ...
void gauss(double A[], int n) {
 double dzielnik;
 int tmp;
 double wiersz_tmp;
 for (int odejmowany=0; odejmowany<n; odejmowany++) {
  for (int i=odejmowany+1; i<n; i++) { // po wierszach bez aktualnie odejmowanego
   if (A[i*n + odejmowany] == 0) {
    tmp = i + 1;
    while (A[n * (tmp++) + odejmowany] == 0);
    if (tmp-- == n) continue;
    //zamieniamy
    for (int j=odejmowany; j<n; j++) {
     wiersz_tmp = A[i*n + j];
     A[i*n + j] = A[tmp*n + j];
     A[tmp*n + j] = wiersz_tmp;
    }
   }
   dzielnik = A[odejmowany*n + odejmowany] / A[i*n + odejmowany];
   for (int j=odejmowany; j<n; j++) { // po kolumnach w aktualnym wierszu, od aktualnie ⤸
```

```
    zerowanej
     A[i*n + j] = A[odejmowany*n +j] - A[i*n + j]*dzielnik;
    }
   }
  }
}

// rozwiazywanie ukladu z macierza trojkatna
void trojkatny(double A[], double b[], double x[], int n) { // jezeli b nie bedzie nam ⤸
potrzebne to mozna wywolywac z x=b
 double suma;

for (int i = n-1; i >= 0; i--) {
   /* x[n-1] = (b[n-1] - a[n-1][n] x[n])/a[n-1][n-1] */

   suma = 0;
   for (int j = i+1; j < n; j++) {
    suma += A[ i*n + j ] * x[j];
//      printf("%d -> %d : %f\n", i, j, A[ i*n + j ]);
   }

   x[i] = (b[i] - suma ) / A[ i*n + i ];
 }
}

// funkcja rozwiazujaca uklad rownan o macierzy trujdiagonalnej (bez zer na diagonali) metoda ⤸
Thomasa
// UWAGA: funkcja niszczy dane wejsciowe
void thomas(double a[], double b[], double c[], double r[], int m) { // liczba elementow ⤸
tablicy b - 1
 int j;

c[0] = c[0] / b[0];
 r[0] = r[0] / b[0];
 for (j=1; j<m; j++) {
  b[j] = b[j] - a[j-1] * c[j-1];
  c[j] = c[j] / b[j];
  r[j] = (r[j] - a[j-1] * r[j-1]) / b[j];
 }
 b[m] = b[m] - a[m-1] * c[m-1];
 r[m] = (r[m] - a[m-1] * r[m-1]) / b[m];
 for (j=m-1; j>=0; j--) {
  r[j] = r[j] - c[j] * r[j+1];
 }
}

// metoda LU
void lu(double A[], int n) {
 int i, j, k;
 double suma;

for (j=0; j<n; j++)
 for (i=0; i<n; i++)
 {
  suma = 0;
  if (i<=j) {
   for (k=0; k<i; k++)
    suma += A [i*n + k] * A [k*n + j];
   A [i*n + j] = A [i*n + j] - suma;
  } else {
```

```
    for (k=0; k<j; k++)
     suma += A [i*n + k] * A [k*n + j];
    A [i*n + j] = (A [i*n + j] - suma) / A [j*n + j];
   }
//    printf("obliczylem element %d %d\n", i, j);
  }
}

// iteracyjna metoda rozwiazywania układów równań
int jacobi(double A[], double b[], double x[], int n, double eps, int limit) {
 int k, j, iteracja = 0;
 bool koniec = false;
 double suma;
 double x_tmp[n];

while (!koniec && iteracja++ < limit) {
  for (j=0; j<n; j++)
   x_tmp[j] = x[j];

  for (j=0; j<n; j++) {
   koniec = true;
   suma = 0;

   for (k=0; k<n; k++)
    if (k != j) suma += x_tmp[k] * A [j*n + k];

   x[j] = ( b[j] - suma ) / A [j*n + j];
   if ( fabs(x_tmp[j] - x[j]) > eps ) koniec = false;
  }
 }
 if (koniec)
  return iteracja;
 else
  return -1;
}

int nadrelaksacja(double A[], double b[], double x[], double omega, int n, double eps, int ⤸
limit) {
 int k, j, iteracja = 0;
 bool koniec = false;
 double suma;
 double x_tmp[n];

while (!koniec && iteracja++ < limit) {
  for (j=0; j<n; j++)
   x_tmp[j] = x[j];

  for (j=0; j<n; j++) {
   koniec = true;
   suma = 0;

   for (k=0; k<j; k++)
     suma += x[k] * A [j*n + k];

   for (k=j; k<n; k++)
     suma += x_tmp[k] * A [j*n + k];

   x[j] = x_tmp[j] + omega * ( b[j] - suma ) / A [j*n + j];
   if ( fabs(x_tmp[j] - x[j]) > eps ) koniec = false;
  }
```

```
 }
 if (koniec)
  return iteracja;
 else
  return -1;
}

int gauss(double A[], double b[], double x[], int n, double eps, int limit) {
 int k, j, iteracja = 0;
 bool koniec = false;
 double suma;
 double x_tmp[n];

while (!koniec && iteracja++ < limit) {
  for (j=0; j<n; j++)
   x_tmp[j] = x[j];

  for (j=0; j<n; j++) {
   koniec = true;
   suma = 0;

   for (k=0; k<j; k++)
     suma += x[k] * A [j*n + k];

   for (k=j+1; k<n; k++)
     suma += x_tmp[k] * A [j*n + k];

   x[j] = ( b[j] - suma ) / A [j*n + j];
   if ( fabs(x_tmp[j] - x[j]) > eps ) koniec = false;
  }
 }
 if (koniec)
  return iteracja;
 else
  return -1;
}
```

TRANSFORMATA_FOURIERA.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje transformatę Fouriera ...
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */

#include <complex>
#include <cmath>

void ft(complex<double> h[], complex<double> H[], int N) {
 int n,k;
```

```
complex<double> w(0, 2*M_PI/N), wn;
 w = exp(w);

for (n=0; n<N; n++) {
  wn = pow(w,n);
  H[n] = 0;
  for (k=0; k<N; k++) {
   H[n] += pow(wn,k) * h[k];
  }
 }
}


void oft(complex<double> H[], complex<double> h[], int N) { // odwrotna ...
 int n,k;

complex<double> w(0, -2*M_PI/N), wn;
 w = exp(w);

for (n=0; n<N; n++) {
  wn = pow(w,n);
  h[n] = 0;
  for (k=0; k<N; k++) {
   h[n] += pow(wn,k) * H[k];
  }
  h[n] /= N;
 }
}
```

SORTOWANIE.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody sortowania
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie ⤸
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */


/*                    */
/*    SORTOWANIE      */
/*                    */

// dane są zdefiniowanym typem struktury zawierającej numeryczne pole klucz oraz pole na ⤸
dane ...

void sort_wybor(dane a[], int m) {
 int i, j, min, min_index;
 dane tmp;

for (j=0; j<m; j++) {
```

```
   // znajdujemny najmniejszy w pozostalej czesci tablicy
   min = a[j].klucz;
   min_index = j;
   for (i=j+1; i<m; i++) {
    if (a[i].klucz < min) {
     min = a[i].klucz;
     min_index = i;
    }
   }
   // wpisujemy najmniejszy na j-ta pozycje
   if (j != min_index) {
    tmp = a[j];
    a[j] = a[min_index];
    a[min_index] = tmp;
   }
  }
 }

void sort_wstaw(dane a[], int m) {
 int i, j=1;
 dane tmp;

         // znajdujemny najmniejszy i wstawiamy na poczatek
                int min_index = 0;

                for (i=1; i<m; i++) {
                        if (a[i].klucz < a[min_index].klucz) {
                                min_index = i;
                        }
                }
                // wpisujemy najmniejszy na j-ta pozycje
                if (0 != min_index) {
                        tmp = a[0];
                        a[0] = a[min_index];
                        a[min_index] = tmp;
                }
  j=2;


for (j; j<m; j++){
  i = j-1;
  tmp = a[j];
  while (tmp.klucz < a[i].klucz) {
   a[i+1] = a[i];
   i--;
  }
  a[i+1] = tmp;
 }
}

void sort_Shell(dane a[], int m) {
 int i, j, h;
        dane tmp;

for (h=1; h<m/9; h=3*h+1);
 for (; h>0; h/=3) {
  // sortowanie przez wstawianie
         for (j=h; j<m; j++){
                 i = j-h;
                 tmp = a[j];
```

```
                while (i>=0 && tmp.klucz < a[i].klucz) {
    a[i+h] = a[i];
                        i-=h;
                }
                if(i!=j)
    a[i+h] = tmp;
        }
 }
}
```

ROZNICZKOWANIE.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody numerycznego różniczkowania funkcji
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */


/*                                    */
/*      RÓŻNICZKOWANIE      */
/*                                    */

// funkcja oblicza pochodna (definicyjnie - ekstrapolujac do zera)
double pochodna(double (*f)(double, double), double k, double x) {
 double g=1, h_tab[5], d_tab[5];
 int i=0;

for (i=0; i<5; i++) {
  h_tab[i] = g;
  //g = g / 2.0;
  g = g / 4.0;
 }

g = f(x,k);

for (i=0; i<5; i++) {
  //d_tab[i] = ( f (x + h_tab[i], k) - g ) / h_tab[i];
  d_tab[i] = ( f (x + h_tab[i], k) - g ) / (2 * h_tab[i]);
 }

//return interpolacja (0, 5, h_tab, d_tab);
 return 2 * interpolacja_Nevil_2 (0, 5, h_tab, d_tab);
}
```

ROWNANIA_ROZNICZKOWE_ZWYCZAJNE.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody numerycznego rozwiązywania równań różniczkowych
 zwyczajnych
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */

typedef void (*func)(double, double*, double*);

void euler (double krok, int ile_krokow, double x, double y[], int n, func f) {
 int i, j;
 double z[n];

// Euler
 for (j=0; j<ile_krokow; j++) {
  f(x, y, z);
  for (i=0; i<n; i++) y[i] = y[i] + krok * z[i];
  x += krok;
 }
}


void midpoint (double krok, int ile_krokow, double x, double y[], int n, func f) {
 int i, j;
 double z[n], z2[n], krok_pol=krok / 2.0;

// punkt srodkowy
 for (j=0; j<ile_krokow; j++) {
  f(x, y, z);

  for (i=0; i<n; i++) z[i] = y[i] + krok_pol * z[i];
  f(x + krok_pol, z, z2);

  for (i=0; i<n; i++) y[i] = y[i] + krok * z2[i];
  x += krok;
 }
}

void heun (double krok, int ile_krokow, double x, double y[], int n, func f) {
 int i, j;
 double z[n], z2[n], z3[n], krok_pol=krok / 2.0;

// Heun
 for (j=0; j<ile_krokow; j++) {
  f(x, y, z);

  for (i=0; i<n; i++) z3[i] = y[i] + krok * z[i];
  f(x + krok, z3, z2);

  for (i=0; i<n; i++) y[i] = y[i] + krok_pol * (z[i] + z2[i]);
  x += krok;
```

```
 }
}

void rk4 (double krok, int ile_krokow, double x, double y[], int n, func f) {
 int i, j;
 double k1[n], k2[n], k3[n], k4[n], tmp[n];

// kolejne kroki
 for (j=0; j<ile_krokow; j++) {
  f(x, y, k1);
  for (i=0; i<n; i++) {
   k1[i] *= krok; // finalny k1

   tmp[i] = y[i] + k1[i] / 2;
  }
  f(x + krok / 2, tmp, k2);
  for (i=0; i<n; i++) {
   k2[i] *= krok; // finalny k2

   tmp[i] = y[i] + k2[i] / 2;
  }
  f(x + krok / 2, tmp, k3);
  for (i=0; i<n; i++) {
   k3[i] *= krok; // finalny k3

   tmp[i] = y[i] + k3[i];
  }
  f(x + krok, tmp, k4);
  for (i=0; i<n; i++) {
   k4[i] *= krok; // finalny k4

   y[i] += k1[i]/6 + k2[i]/3 + k3[i]/3 + k4[i]/6;
  }

  x += krok;
 }
}

#define EPSILON 0.00000001
void rk4_adapt (double krok, int ile_krokow, double x, double y[], int n, func f) {
 int i, j;
 double k1[n], k2[n], k3[n], k4[n], tmp[n], y1[n], y2[n];
 double Z, pol_krok, x_max = krok*ile_krokow; // dla zachowania kompatybilnosci wstecznej
 postaci wywolania funkcji

// kolejne kroki
 while (x <= x_max) {
     // dla CALEGO kroku
  f(x, y, k1);
  for (i=0; i<n; i++) {
   k1[i] *= krok; // finalny k1

   tmp[i] = y[i] + k1[i] / 2;
  }
  f(x + krok / 2.0, tmp, k2);
  for (i=0; i<n; i++) {
   k2[i] *= krok; // finalny k2

   tmp[i] = y[i] + k2[i] / 2;
  }
```

```
f(x + krok / 2.0, tmp, k3);
for (i=0; i<n; i++) {
 k3[i] *= krok; // finalny k3

 tmp[i] = y[i] + k3[i];
}
f(x + krok, tmp, k4);
for (i=0; i<n; i++) {
 k4[i] *= krok; // finalny k4

 y1[i] = y[i] + k1[i]/6 + k2[i]/3 + k3[i]/3 + k4[i]/6;
}

  // dla POL kroku
pol_krok = krok/2.0;
// pierwszy raz
f(x, y, k1);
for (i=0; i<n; i++) {
 k1[i] *= pol_krok; // finalny k1

 tmp[i] = y[i] + k1[i] / 2;
}
f(x + pol_krok / 2.0, tmp, k2);
for (i=0; i<n; i++) {
 k2[i] *= pol_krok; // finalny k2

 tmp[i] = y[i] + k2[i] / 2;
}
f(x + pol_krok / 2.0, tmp, k3);
for (i=0; i<n; i++) {
 k3[i] *= pol_krok; // finalny k3

 tmp[i] = y[i] + k3[i];
}
f(x + pol_krok, tmp, k4);
for (i=0; i<n; i++) {
 k4[i] *= pol_krok; // finalny k4

 y2[i] = y[i] + k1[i]/6 + k2[i]/3 + k3[i]/3 + k4[i]/6;
}
// drugi raz
f(x + pol_krok, y2, k1);
for (i=0; i<n; i++) {
 k1[i] *= pol_krok; // finalny k1

 tmp[i] = y2[i] + k1[i] / 2;
}
f(x + pol_krok + pol_krok / 2.0, tmp, k2);
for (i=0; i<n; i++) {
 k2[i] *= pol_krok; // finalny k2

 tmp[i] = y2[i] + k2[i] / 2;
}
f(x + pol_krok + pol_krok / 2.0, tmp, k3);
for (i=0; i<n; i++) {
 k3[i] *= pol_krok; // finalny k3

 tmp[i] = y2[i] + k3[i];
}
f(x + krok, tmp, k4);
```

```
  for (i=0; i<n; i++) {
   k4[i] *= pol_krok; // finalny k4

   y2[i] = y2[i] + k1[i]/6 + k2[i]/3 + k3[i]/3 + k4[i]/6;
  }

/*    for (i=0; i<n; i++) y2[i]=y[i];
  rk4 (krok / 2.0, 2, x, y2, n, f);
  for (i=0; i<n; i++) y1[i]=y[i];
  rk4 (krok, 1, x, y1, n, f);*/

    // sprawdzam czy krok byl dobry i podejmuje odpowiednie dzialania ...
  // obliczam Z
  Z = 0;
  for(i=0; i<n; i++) Z += pow(y1[i]-y2[i] , 2);
  Z = pow(16.0/15.0 * sqrt(Z) / EPSILON , 1.0/5.0);

  if (Z > 3) {
   krok *= 2 / Z;
  } else {
   for(i=0; i<n; i++) y[i]=y1[i];
   x += krok;
   krok *= 2 / Z;

  #ifdef _WYDRUK_WYNIKOW_POSREDNICH_
   _WYDRUK_WYNIKOW_POSREDNICH_
  #endif

  }
//printf("x=%f\tZ=%f\tkrok(nowy)=%f\ty1=%f\ty2=%f\n", x, Z, krok, y1[0], y2[0]);
 }
}

void adams2 (double krok, int ile_krokow, double x, double y[], int n, func f) {
 int i, j;
 double z1[n], z2[n], krok_pol=krok / 2.0;

// z innej metody pozyskujemy 2 startowe z
 heun (krok, 1, x, y, n, f); f(x, y, z2); x += krok;
 heun (krok, 1, x, y, n, f); f(x, y, z1); x += krok;
 ile_krokow -= 2;

// kolejne kroki
 for (j=0; j<ile_krokow; j++) {
  for (i=0; i<n; i++) {
   y[i] += krok_pol * (3*z1[i] - z2[i]); // z1 = f[n], z2 = f[n-1]
   z2[i] = z1[i];
  }
  f(x, y, z1); // generujemy nowy z1
  x += krok;
 }
}

void adams4 (double krok, int ile_krokow, double x, double y[], int n, func f) {
 int i, j;
 double z1[n], z2[n], z3[n], z4[n], krok_pol=krok / 24.0;

// z innej metody pozyskujemy 4 startowe z
 rk4 (krok, 1, x, y, n, f); f(x, y, z4); x += krok;
 rk4 (krok, 1, x, y, n, f); f(x, y, z3); x += krok;
```

```
 rk4 (krok, 1, x, y, n, f); f(x, y, z2); x += krok;
 rk4 (krok, 1, x, y, n, f); f(x, y, z1); x += krok;
 ile_krokow -= 4;

// kolejne kroki
 for (j=0; j<ile_krokow; j++) {
  for (i=0; i<n; i++) {
   y[i] += krok_pol * (55*z1[i] - 59*z2[i] + 37*z3[i] - 9*z4[i]); // z = f[n], z2 = f[n-1], ..
   .
   z4[i] = z3[i];
   z3[i] = z2[i];
   z2[i] = z1[i];
  }
  f(x, y, z1); // generujemy nowy z1
  x += krok;
 }
}

void pk (double krok, int ile_krokow, double x, double y[], int n, func f) { // predyktor
korektor
 int i, j, k;
 double y_org[n], z1[n], z2[n], z3[n], krok_pol = krok/12.0;

// kolejne kroki
 for (j=0; j<ile_krokow; j++) {
  for (i=0; i<n; i++) y_org[i] = y[i];
  //adams2 (krok, 1, x, y_jaw, n, f) - PREDYKTOR

   // z innej metody pozyskujemy 2 startowe z
   heun (krok, 1, x, y, n, f); f(x, y, z2); x += krok;
   heun (krok, 1, x, y, n, f); f(x, y, z1); x += krok;

   for (i=0; i<n; i++) y[i] += krok/2.0 * (3*z1[i] - z2[i]); // z1 = f[n], z2 = f[n-1]

   for (k=0; k<3; k++) {
    f(x, y, z3); // generujemy z3 = f[n+1]
  // A-M rz 3 - KOREKTOR
    for (i=0; i<n; i++) y[i] = y_org[i] + krok_pol * (5*z3[i] + 8*z1[i] - z2[i]);
   }
 }
}
```

ROWNANIA_ROZNICZKOWE_CZASTKOWE.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody numerycznego rozwiązywania równań różniczkowych
 cząstkowych
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */
```

```c
#include "alg_thomasa.c"

void ftcs(double T[], int ilosc_x, int ilosc_t, double krok_x, double krok_t) {
 int i, j;

double r = krok_t/(krok_x * krok_x);

//pierwszy wiersz mamy z warunku brzegowego
 for (i=1; i<=ilosc_t; i++) {
  //piersza i ostatnia kolumne mamy z war. brzegowego
  for (j=1; j<ilosc_x; j++) {

   T [ i*ilosc_x + j ] = \
    r * T [ (i-1)*ilosc_x + j-1 ] + \
    (1-2*r) * T [ (i-1)*ilosc_x + j ] + \
    r * T [ (i-1)*ilosc_x + j+1 ];

  }
 }
}

void ftcs_mem(double T[], int ilosc_x, int ilosc_t, double krok_x, double krok_t) {
 int i, j;

double r = krok_t/(krok_x * krok_x);
 double T_poprzedni, T_poprzedni_tmp;

//pierwszy wiersz mamy z warunku brzegowego
 for (i=1; i<=ilosc_t; i++) {
  //piersza i ostatnia kolumne mamy z war. brzegowego
  T_poprzedni=T[0];
  for (j=1; j<ilosc_x; j++) {
   T_poprzedni_tmp=T[j];

   T [ j ] = \
    r * T_poprzedni + \
    (1-2*r) * T [ j ] + \
    r * T [ j+1 ];

   T_poprzedni=T_poprzedni_tmp;
  }
 }
}

void ftcs_mem2(double T[], double L[], double P[],  int ilosc_x, int ilosc_t, double krok_x, ⤸
double krok_t) {
 int i, j;

double r = krok_t/(krok_x * krok_x);
 double T_poprzedni, T_poprzedni_tmp;

//pierwszy wiersz mamy z warunku brzegowego
 for (i=1; i<=ilosc_t; i++) {
  //piersza i ostatnia kolumne mamy z war. brzegowego
  T_poprzedni = L[i-1];
  T [ ilosc_x ] = P[i-1];

  for (j=1; j<ilosc_x; j++) {
   T_poprzedni_tmp=T[j];
```

```
   T [ j ] = \
    r * T_poprzedni + \
    (1-2*r) * T [ j ] + \
    r * T [ j+1 ];

   T_poprzedni=T_poprzedni_tmp;
  }
 }
}

void cn(double T[], double L[], double P[],  int ilosc_x, int ilosc_t, double krok_x, double ⤦
krok_t) {
 int i, j;

double A, B, A2, B2, T_poprz, T_poprz_tmp, r[ ilosc_x ], b [ ilosc_x ];

A = krok_t/(krok_x * krok_x);
 B = 1+A;
 A2 = A/2.0;
 B2 = 1-A;
 A = -A/2.0;

for (i=1; i<=ilosc_t; i++) {
  T_poprz=T[0];
  for (j=1; j<ilosc_x; j++) {
   T_poprz_tmp = T[j];
   T[j] = A2 * (T_poprz + T[j+1]) + B2 * T[j];
   T_poprz=T_poprz_tmp;
   b[j] = B;
  }
  T[1] -= A*L[j-1];
  T[ilosc_x-1] -= A*P[j-1];

  T[0]=L[j-1]; T[ilosc_x]=P[j-1]; // dla nastepnego kroku

//TEST
//b[1]=2; b[2]=2; A=4; T[1]=2; T[2]=10; ilosc_x=3;
for (j=0;j<=ilosc_x;j++) printf(" %f * ", T[j]);
printf("\n\n");
  // rozwiazujemy uklad rownan
  T[1] /= b[1];
  for (j=2; j<ilosc_x; j++) {
   b[j] -= A * A / b[j-1];
   T[j] = (T[j] - A*T[j-1]) / b[j];
  }
  for (j=ilosc_x-1; j>0; j--) {
   T[j] = T[j] - A*T[j+1]/b[j];
  }
 }
}
```

MIEJSCA_ZEROWE.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody poszukiwania miejsc zerowych
 * (czyli numerycznego rozwiązywania równań ...)
```

```
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie ⤸
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */


/*                                        */
/*      ZNAJDOWANIE MIEJSC ZEROWYCH       */
/*                                        */

// funkcja znajduje miejsca zerowe metodą bisekcji (szukania punktu zmiany znaku)
double miejsca_zerowe_bisekcja(double (*f)(double, double), double k, double x1, double x2, ⤸
double eps) {
 double x, f1, f2, fx;

if( f(x1, k) * f(x2, k) > 0 ) {
  printf("Blaedne dane\n");
  return sqrt(-2);
 }

while ( fabs(f(x, k)) > 0 ) {
  f1 = f(x1, k);
  f2 = f(x2, k);
  x = (x1 + x2) / 2.0;
  fx = f(x, k);

  if (fabs(fx) < eps) {
   return x;
  } else if (fx*f1 > 0) {
   x1 = x;
  } else if (fx*f2 > 0) {
   x2 = x;
  }
 }
}

// funkcja znajduje miejsca zerowe metodą pochodnej (Newtona)
double miejsca_zerowe_Newton(double (*f)(double, double), double (*df)(double, double), ⤸
double k, double x, double eps) {
 while( fabs(f(x, k)) > eps ) {
  x = x - f(x,k) / df(x,k);

}

return x;
}

// funkcja znajduje miejsca zerowe metodą siecznych
double miejsca_zerowe_sieczne(double (*f)(double, double), double k, double x1, double x2, ⤸
double eps) {
 double x, fx, f1, f2;

f1 = f(x1, k);
 f2 = f(x2, k);

do {
```

```
  x = x1 - f1 * (x2 - x1) / (f2 - f1);
  fx = f(x, k);

  if (f1*fx>0) {
   x2 = x;
   f2 = fx;
  } else {
   x1 = x;
   f1 = fx;
  }
 } while( fabs(f(x, k)) > eps );

return x;
}
```

INTERPOLACJA.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody interpolacji
 * (obliczania wartosci dla zadanego punktu gdy mamy wartosci w innych punktach)
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie ⤸
 zagwarantować,
 * że działają one w pełni poprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */


/*                                          */
/*      INTERPOLACJA WIELOMIANEM            */
/*                                          */

double interpolacja (double x, int STOPIEN, double tab_x[], double tab_f[]) {
 double wynik = 0, L;
 for (int i=0; i<STOPIEN; i++) {
  L = 1;
         for (int j=0; j<STOPIEN; j++) {
                 if (j != i)
                         L = L * (x - tab_x[j]) / (tab_x[i] - tab_x[j]);
         }
  wynik = wynik + tab_f[i]*L;
 }

return wynik;
}

// Nevil (wersja 1)
// algorytm Nevila (niezaleznie od wersji implementacji) jest bardzo dobry do ekstrapolacji w ⤸
zerze
double interpolacja_Nevil_1 (double x, int STOPIEN, double tab_x[], double tab_f[]) {
 double P [STOPIEN] [STOPIEN]; //P[i][j]
 int i,j;
```

```
for (i=0; i < STOPIEN; i++){
  P[i][0] = tab_f[i];
 }

for (j=1; j < STOPIEN; j++){
  for (i=j; i < STOPIEN; i++){
   //P[i][j] = f ( P[i-1][j-1], P[i][j-1] );
   P[i][j] = ( (x - tab_x[i]) * P[i-1][j-1] + (tab_x[i-j] - x) *  P[i][j-1] ) / (tab_x[i-j] - ⤸
   tab_x[i]);
  }
 }

return P [STOPIEN-1] [STOPIEN-1];
}

// Nevil (wersja 2 - udoskonalona operująca na mniejszej tablicy)
double interpolacja_Nevil_2 (double x, int STOPIEN, double tab_x[], double tab_f[]) {
 double P[STOPIEN];

for (int i=0; i < STOPIEN; i++) {
  P[i] = tab_f[i];
  for (int j=i-1; j>=0; j--) {
   // printf("MAMY i = %d\t oraz j = %d a nasze x[i] = %e\t x[i-j] = %e\n",i,j, tab_x[i], ⤸
   tab_x[j]);
   P[j] = ( (x - tab_x[i]) * P[j] + (tab_x[j] - x) *  P[j+1] ) / (tab_x[j] - tab_x[i]);
  }
 }

return P [0];
}

// double tab_f[STOPIEN]; double tab_x[STOPIEN]; double f[STOPIEN*STOPIEN];
// wynik - wspolczynniki (tab_a dla poniższego algorytmu Hornera) to pierwsze STOPIEN ⤸
elementów z (pseudo dwuwymiarowej) tablicy f
void interpolacja_Newton (int STOPIEN, double tab_x[], double tab_f[], double *f) {
 int i,j;

for (i=0; i < STOPIEN; i++){
  f[i*STOPIEN] = tab_f[i];
 }

for (j=1; j < STOPIEN; j++){
  for (i=0; i < STOPIEN-j; i++){
   f[i*STOPIEN+j] = ( f[(i+1)*STOPIEN+j-1] - f[i*STOPIEN+j-1] ) / ( tab_x[i+j] - tab_x[i] );
  }
 }
}

// funkcja zwraca wartosc wielomianu obliczoną metodą Hornera: a_0 + (x - x_1) * (a_1 + (x -⤸
x_2) * (a_2 + (x - x_3) * (a_4 + ... )))
double Horner (int STOPIEN, double x, double tab_x[], double tab_a[]) {
 double wynik=0;

for (int i=STOPIEN-1; i>0; i--){
  wynik = (tab_a[i] + wynik)*(x-tab_x[i-1]);
 }
 wynik = wynik + tab_a[0];

return wynik;
}
```

```
/*                                              */
/*      DOPASOWANIE GLADKIE - SPLINE             */
/*                                              */

// plik z funkcja rozwiazujaca uklad rownan o macierzy trujdiagonalnej (bez zer na diagonali) ⤸
metoda Thomasa
// UWAGA: program niszczy dane wejsciowe
void Thomas(double a[], double b[], double c[], double r[], int m) { // m = liczba elementow ⤸
tablicy b - 1
 int j;

c[0] = c[0] / b[0];
 r[0] = r[0] / b[0];
 for (j=1; j<m; j++) {
  b[j] = b[j] - a[j-1] * c[j-1];
  c[j] = c[j] / b[j];
  r[j] = (r[j] - a[j-1] * r[j-1]) / b[j];
 }
 b[m] = b[m] - a[m-1] * c[m-1];
 r[m] = (r[m] - a[m-1] * r[m-1]) / b[m];
 for (j=m-1; j>=0; j--) {
  r[j] = r[j] - c[j] * r[j+1];
 }
}

// wymaga wczesniejszego zdeklarowania i przekazania tablic: double a[n-1], b[n], c[n-1];
// gdzie n - ilosc danych wejsciowych
void wspolczynniki_spline (double x_tab[], double y_tab[], double a[], double b[], double c[],⤸
int n) {
 // przygotowanie dla alg. Thomasa
 double r[n], h[n];
 int i;

h[0] = 0;
 for (i=1; i<n; i++) h[i] = x_tab[i] - x_tab[i-1];

for (i=0; i<n-2; i++) a[i] = h[i+1] / (h[i+1] + h[i+2]);

for (i=0; i<n; i++) b[i] = 2;

c[0] = 0;
 for (i=1; i<n-2; i++) c[i] = h[i+1] / (h[i] + h[i+1]);

r[0]=0; r[n-1]=0;
 for (i=1; i<n-1; i++) r[i]=((y_tab[i+1]-y_tab[i]) / h[i+1] - (y_tab[i]-y_tab[i-1]) / h[i]) * ⤸
 6 / (h[i] + h[i+1]);

// alg. Thomasa
 Thomas(a, b, c, r, n-1);

// wyznaczenie parametrow
 for (i=0; i<n-1; i++) {
  a[i] = (r[i+1] - r[i]) / (6 * h[i+1]);
  b[i] = r[i] / 2;
  c[i] = (y_tab[i+1] - y_tab[i]) / h[i+1]  -  h[i+1] * (2*r[i] + r[i+1]) / 6;
 }
```

```
}

// obliczamy wartosci wielomianu z wspolczynnikami a[i], b[i], c[i] , d[i]
// dla zadanego x (bierzemy stowane wspolczynniki, gdy x nalezy do i-tego odcinka
// uzyskujemy aproksymowaną wartosc y(x)
void spline (double x_tab[], double y_tab[], double a[], double b[], double c[], double x) {
 if (x >= x_tab[0] && x <= x_tab[n-1]) {
  i = 1;
  while (x > x_tab[i]) i++;
  // x nalezy do przedzialu x[i-1] do x[i]
  i--;

  // wynik dla zadanego x
  return y_tab[i] + ( c[i] + ( b[i] + a[i] * (x-x_tab[i]) ) * (x-x_tab[i]) ) * (x-x_tab[i]);
 }
}
```

CALKOWANIE.CC

```
/*
 * plik ma na celu prezentację podstawowych zagadnień z metod numerycznych (dla C / C++)
 *
 * prezentuje podstawowe metody numerycznego całkowania funkcji
 *
 * Plik stanowi część zbióru funkcji prezentujących podstawy metod numerycznych.
 * Jako że programy te napisałem pewien czas temu nie jestem w stanie obecnie
 zagwarantować,
 * że działają one w pełni proprawnie. Zachęcam do zapoznania się z skryptem z metod
 * numerycznych dostępnym na stronie http://www.fuw.edu.pl/~pliszka/mn/
 *
 */

/*                                */
/*     CALKOWANIE - GAUSS          */
/*                                */

// deklaracje dla n = 4
static double wezel4[] = {0.3399810435, -0.3399810435, 0.8611363115, -0.8611363115};
static double waga4[] = {0.6521451548, 0.6521451548, 0.3478548451, 0.3478548451};

// deklaracje dla n = 2
static double wezel2[] = {0.5773502691, -0.5773502691};
static double waga2[2] = {1.0, 1.0};

// deklaracje uniwersalne
static double *wezly[] = {0, 0, wezel2, 0 , wezel4};
static double *wagi[] = {0, 0, waga2, 0, waga4};

double calka_Gauss (double (*f)(double, double), double a, double b, int n, double k) {

if (n != 2 && n != 4) {
  return 0; // BLAD
 }

double I=0;

for (int i=0; i<n; i++) {
  I = I + wagi[n][i] * f( (b-a) * wezly[n][i] / 2 + (a+b)/2 , k);
```

```
 }

I = I * (b-a) / 2;

return I;
}

// rekurencyjne "precyzjowanie" obliczonej calki (gdy maga dokladnosc na danym odcinku badamy ↩
go poprzez podzial na coraz mniejsze)
// korzystajace z powyzszej funkcji: calka_Gauss
double calka_Gauss_rekurencja (double (*f)(double, double), double a, double b, double k, ↩
double eps, int stop) {

double cal = gauss(f, a, b, stop, k), cal_1 = gauss(f, a, (a+b)/2.0, stop, k), cal_2 = gauss(↩
 f, (a+b)/2.0, b, stop, k);

        if ( (fabs(cal) < fabs(cal_1 + cal_2) + eps) && (fabs(cal) > fabs(cal_1 + cal_2) - ↩
        eps) ) {
                return cal;
        } else {
                return calka(f, a, (a+b)/2, k, eps, stop) + calka(f, (a+b)/2, b, k, eps, stop)↩
                ;
        }
}



/*                                  */
/*      CALKOWANIE -                 */
/*                                  */
static int sigma_1[] = {1, 1};
static int sigma_2[] = {1, 4, 1};
static int sigma_3[] = {1, 3, 3, 1};
static int sigma_4[] = {7, 32, 12, 32, 7};
static int sigma_5[] = {19, 75, 50, 50, 75, 19};
static int sigma_6[] = {41, 216, 27, 272, 27, 216, 41};

// deklaracje uniwersalne
static int *sigmy[] = {0, sigma_1, sigma_2, sigma_3, sigma_4, sigma_5, sigma_6};
static int ns[] = {0, 2, 6, 8, 90, 288, 840};

double calkowanie(double f_i[], double a, double b, int n) {
 double suma = 0;
 for (int i=0; i<=n; i++) {
  suma = suma + (double)sigmy[n][i] * f_i[i];
 }

double wynik;
 wynik = suma * (b-a) / ns[n];

return wynik;
}

double calka(double (*f)(double, double), double a, double b, double k, double eps) {
 double f_i[2];
 double s = 0, s_old=0, nowe, x;

for (int i=0; i<=1; i++) {
  f_i[i] = f( a + i * (b-a) , k );
 }
```

```
 s = calkowanie (f_i, a, b, 1);

int iteracja = 0;
 while (fabs(s) < eps || fabs(s) > fabs(s_old) + eps || fabs(s) < fabs(s_old) - eps ) {
  iteracja++;
  s_old = s;

  nowe = 0;
  x = a + (b-a)/(2*iteracja);
  for (int i=0; i<iteracja; i++) {
   nowe = nowe + f(x, k);
   x = x + (b-a) / iteracja;
  }
  s = s / 2  +  (b-a) / (2*iteracja) * nowe;
 }
 return s;
}

// Romberg - ekstrapolacja z kilku iteracji powyzszej funkcji z h -> 0
double romberg(double (*f)(double, double), double a, double b, double k, double eps) {
        double f_i[2];
 double s_tab[5], h_tab[5];
 double s = 0, nowe, x;

for (int i=0; i<=1; i++) {
  f_i[i] = f( a + i * (b-a) , k );
 }
 s = calkowanie (f_i, a, b, 1);


for (int iteracja=0; iteracja<5; iteracja++) {
  nowe = 0;
  x = a + (b-a)/(2*iteracja);
  for (int i=0; i<iteracja; i++) {
   nowe = nowe + f(x, k);
   x = x + (b-a) / iteracja;
  }
  s = s / 2  +  (b-a) / (2*iteracja) * nowe;
  s_tab[iteracja] = s;
  h_tab[iteracja] = (b-a) / (2*iteracja);
 }

// a teraz ekstrapolacja s prz h dazacym do zera
 // korzystamy do tego z algorytmu Nevilla

return interpolacja_Nevil_2 (0, 5, h_tab, s_tab);
}
```

Linki i moje projekty

Zachęcam także do zapoznania się z moimi projektami związanymi z językiem C/C++: Sterowanie magnetowidem[1034], detekcja radiometeorów, powiadamiacz o IM dla jabberd2 (serwer - mysql, autoryzacja PAM, parsowanie napisów, klient - AVR NutOS), sterownik ogrzewania (AVR), sterownik oświetlenia (AVR).

Zobacz w Sieci: C++ bez cholesterolu[1035], KDMwiki - Programowanie[1036] (głównie o uruwnoleglaniu obliczeń), Serwis Numeryczny ICM[1037], Wiki o C/C++[1038], niekompatybilności między C a C++[1039], libftdi[1040], libmodbus[1041], Optimizing C++[1042], C++ Programming As A Set Of Problems[1043].

Programowanie w Bash'u

Praca w trybie wsadowym polega na wykonywaniu zapisanego w pliku tekstowym ciągu poleceń, dlatego też przed rozpoczęciem programowania warto zapoznać się z działaniem basha jako powłoki systemowej. Istnieje kilka metod wykonania takiego pliku:

- `. sciezka_do_pliku` lub >`source sciezka_do_pliku` - polecenia zwarte w podanym pliku wykonane zostaną przez bieżącą instancję basha (przydatne do włączania plików z zmiennymi konfiguracyjnymi, aby wczytać tylko zmienne z pliku (bez wykonywania zawartych w nim poleceń) możemy skorzystać z następującego kodu `` eval export `cat config.sh` ``, wczyta on nawet prawie poprawnie wieloliniowe zmienne z podanego pliku - nowa linia zostanie zastąpiona spacją)
- `bash sciezka_do_pliku` - zostanie uruchomiony osobny proces powłoki który wykona polecenia zwarte w podanym pliku
- `sciezka_do_pliku` (w przypadku gdy jest w bieżącym katalogu to rozpoczęta od `./`) - jak wyżej z tym że plik musi mieć prawa wykonywalności i zazwyczaj rozpoczyna się (pierwsza linijka) od określenia interpretatora w przypadku Bash'a będzie to `#!/bin/bash` (uwaga na niektórych systemach bash może być też albo tylko (jak w BeOSie) określany przez ścieżkę /bin/sh); jeżeli taki plik zostanie umieszczony w katalogu będącym w ścieżce wyszukiwania (zmienna $PATH) to może być odpalany po nazwie

Skrypty często mają rozszerzenia `.sh`, jednak także często odchodzi się od tego aby skrypt wywoływany był w oparciu o samą nazwę (tak jak dowolna inna komenda).

Pisanie skryptów bashowych jest w zasadzie typowym programowaniem (i wbrew pozom ten język programowania ma ogromne możliwości i wiele zagadnień można rozwiązać w nim bardzo łatwo i szybko). Program taki korzysta z dowolnych instrukcji powłoki (cd, ...) oraz wywołań innych programów (cp, rm, gawk, grep, ...). W śród wywoływanych zewnętrznych programów na szczególną miejsce zasłużył sobie (g)awk, będący interpretowanym językiem programowania służącym do przetwarzania tekstów. Z kolei wśród poleceń wbudowanych powłoki jest bardzo wiele instrukcji przeznaczonych głównie dla skryptów (aczkolwiek każdą z nich można wykorzystać w trybie interaktywnym). W tym miejscu warto także wspomnieć o pakietach takich jak `zenity`, `kdialog`, `gdialog` umożliwiających wyświetlenie z poziomu skryptu okienek dialogowych środowiska graficznego (X'ów) a także o dcop umożliwiającym sterowanie programami KDE z powłoki. Przydatne może być też określenie poleceń wykonywanych gdy następuje przerwanie wykonywania skryptu (np. Ctrl+C lub kill -15) - `trap "{ echo "to koniec"; }" EXIT`.

Zmienne

Jak w każdym przyzwoitym języku programowania możemy korzystać ze zmiennych.

Na wstępie warto wspomnieć o paru parametrach (jest ich trochę więcej) do których dostęp mamy prze $:

- 0 - nazwa skryptu lub powłoki
- liczba - odpowiedni argument wywołania (parametr pozycyjny)
- * - kolejne parametry pozycyjne, równoważne "$1 $2 ..."
- @ - kolejne parametry pozycyjne, równoważne "$1" "$2" ...
- # - liczba parametrów pozycyjnych
- ? - kod zakończenia ostatniej instrukcji

Prawie wszystkie zmienne w bashu mają zasięg globalny. Wyjątkiem są zmienne odpowiedzialne za przechowywanie argumentów skryptu / funkcji (każda funkcja ma swój niezależny zestaw i nie ma dostępu do argumentów skryptu) oraz zmienne definiowane przy pomocy słowa kluczowego local przed nazwą zmiennej. Kolejnym wyjątkiem są zmienne przekazywane do podprocesów - są one w nich dostępne, ale ich zmiany nie są widoczne w głównym skrypcie - np. (odkomentowanie linii z ps pokazuje dlaczego tak się dzieje):

```
zm=1
#ps x -l | tail
echo -e "ttt\nooo" | while read f; do
        zm=$(( $zm + 1 ))
        echo " $zm"
        #ps x -l | tail
done
echo $zm
```

Dostępne są też zmienne środowiskowe - takie jak $HOME czy też $PATH, a także np. pseudozmienna $RANDOM zwracająca losową liczbę. Przydatna jest także komenda `eval` wykonująca komendę powstałą z zczytania przekazanych do niej argumentów (w tym zmiennych) - dzięki jej użyciu możemy np. część konstrukcji case, if czy jakiejś pętli przechowywać w zmiennej (np. `LISTA_WYBORU="a) echo AA;; b) echo BB;;"; eval case b in $LISTA_WYBORU esac`).

Możliwe jest także (na co najmniej 3 sposoby) wydobycie zawartości zmiennej, której nazwę mamy w innej zmiennej:

```
A="to chcemy wyświetlić"; B=A;
# metoda pierwsza
C=${!B}; echo $C
# metoda druga
C='eval "echo \$$B"'; D=`eval "$C"`; echo $D
# metoda trzecia
C=$(C='eval "echo \$$B"'; eval $C); echo $C
```

```
#!/bin/bash

# to jest komentarz

ala="ma kota"
wiek_kota=5
# powyzej zdefiniowalismy dwie zmienne, wazne aby miedzy nazwa a = nie bylo spacji

echo "Ala $ala, który ma $wiek_kota lat"
# oraz podstawiliśmy je - służy do tego operator $

# tutaj też uwaga odnośnie różnych cudzysłowów:
# "" - zmienne są podstawiane itp, '' - tekst nie jest modyfikowany,
# `` - służy do wykonania polecenia, którego stdout może być zapisany w zmiennej
#      w Bashu takie samo działanie ma $() ta metoda może być zagnierzdzana
# (jak w poniższym przykładzie):
wynik=`ls $HOME`
# tutaj widzimy też że w ten sam sposób możemy obsługiwać zmienne środowiskowe ...
# warto także zaznaczyć że aby zmienić zmienne środowiskowe tak aby zmiany były
# widoczne poza skryptem po   nadaniu nowej wartości zmiennej nalezy skorzystać z
# komendy: export $ZMIENNA

echo "${ala}makota"
# klamerkami zaznaczyliśmy co jest nazwa zmiennej a co napisem ...


echo ${nie_ma_takiego:-"Hello World"}
# wypisze $nie_ma_takiego gdy ustawiony i niepusty albo (w przeciwnym razie) "Hello World"
# gdy zamiast :- użyjemy := dodatkowo zmienna nie_ma_takiego zostanie ustawiona na "Hello ⤸
World"

# jest jeszcze pare ciekawych zastosowań ${} ...

echo ${ala:+"Hello World"}
# podobnie jak powyżej, ale wypisze "Hello World" gdy $ala jest zdefiniowana (nie pusta)
# w przeciwnym razie użyje napisu pustego,

# gdy w powyższych pominiemy : zdeklarowany napis pusty będzie rozróżniany od wartości ⤸
niezdeklarowanej


echo ${#ala}
# wypisze długość napisu w zmiennej ala

abc="abcdefgh"
```

```
echo ${abc#"ab"} ${abc%"h"} ${abc:1} ${abc:2:2}
# wypisze $abc:
# z odrzuconym początkowym ab
# końcowym h
# od pozycji 1 (pierwsza litera to pozycja 0)
# 2 litery od pozycji 2

${abc:0:$(( ${#abc} - 1 ))}
# powyższa dziwna konstrukcja służy wypisaniu zawartości zmiennej bez ostatniego znaku

abc="abcdeabcf"
echo ${abc/ab/AB} ${abc//ab/AB}
# wypisze $abc z zastąpionym pierwszym (a potem wszystkimi) wystąpieniami ab przez AB
# jeżeli napis dopasowywany rozpoczniemy od # musi on być na początku zmiennej
# jeżeli od % musi być na końcu

# podobne rzeczy można robić korzystając z expr - w szczególności
# expr match $zmienna 'wyrazenieregularne1\(podwyrazenie\)wyrazenieregularne2'
# zwróci część podanej zmiennej pasującej do podanego podwyrażenia ujętego w \( \)
expr $abc : '.*\([^df]*\)'


# możemy używać $ wewnątrz napisów - w tym celu zabezpieczmy go \ lub napis umieszczmyu
w "
xx='${ala} ma $HOME'
ala="ALA"
export ala
echo $xx
# jak widać zmienne wtedy nie są podmieniane, ale jeżeli wyeksportujemy zmienną
# i przekażemy taki napis do envsubst to zostaną podstawione
echo $xx | envsubst
# możemy nawet określić które mają być podstawiane ...
echo $xx | envsubst '$ala'


# ale to jeszcze nie wszystko bash umożliwia nam korzystanie z TABLIC:
tablica[0]="to jest element 0"
tablica[1]="to jest element 1"
element=1
echo ${tablica[1]} " --- " ${tablica[0]}

# @ i * zachowują się tak jak w $@ i w $* ...
echo ${tablica[@]}

echo ${#tablica[@]}
# zwróci ilość elementów w tablicy, ale
echo ${#tablica[1]}
# zwróci długość elementu o indeksie 1

# pozostałe operacje napisowe gdy są stosowane do ${tablica[@]} stosują się do
# całej zawartości tablicy a gdy ${tablica[$i]} do elementu określonego w $i ...
```

Pętle i instrukcje wyboru

Do dyspozycji mamy także pętle oraz instrukcje wyboru. W śród pętli warto zwrócić uwagę iż mogą one funkcjonować w oparciu o warunek zawarty w nawiasach kwadratowych lub też na liście plików / słów, czy też do czasu aż jakaś komenda się wykonuje.

```
#!/bin/bash

for nazwa in /tmp/* ; do
        echo $nazwa;
done
# powyzsza petla wypisze nazwy wszystkich plików znajdujących się w /tmp

for (( i=0 ; $i<=20 ; i++ )) ; do
        echo $i;
done
# powyzsza petla wypisze liczby od 0 do 20

cat /etc/fstab | while read slowo reszta_linii; do
        echo $reszta_linii;
done
# powyższa pętla wypisze po kolei wszystkie wiersze pliku przekazanego przez stdin
# (cat nazwa_pliku |) z pominięciem pierwszego słowa (pierwsze słowo wczytywane było
# do zmiennej slowo, której nie wypisywaliśmy) jest też pętla until która odpowiada
# while z zanegowanym warunkiem (w tym wypadku warunkiem jest instrukcja read)
```

Dość praktycznym zastosowaniem pętli w bashu jest wywoływanie czegoś dla każdej linii wejścia - np. `grep -r 'warunek'* | cut -f1 -d':'| uniq |whileread f ;doecho $f; sed 's#szukany#zamiennik#g'"$f" > /tmp/zam; mv -f /tmp/zam "$f";done`. Kod ten wyszukuje rekurencyjnie pliki zawierające ”warunek” oraz zamienia w nich wszystkie wystąpienia ”szukany” na ”zammiennik”. Warto tutaj zwrócić uwagę na wykorzystanie pętli while oraz jednolinijkowy zapis (wygodny przy pracy interaktywnej z bashem).

Więcej informacji o warunkach jakie możemy zawrzeć w [] wykorzystywanym zarówno w pętlach jak i warunkach typu if patrz: `man 1 test`.

```
#!/bin/bash

# ponizszy wpis wlacza wypisywanie wykonywanych polecen
#- bardzo przydatne przy debugowaniu skryptow ...
set -x

COS="ryba";

# poniższy fragment obrazuje instrukcje case na prostym przykładzie
case $COS in
        kot | pies)
                echo "kot lub pies"
                ;;
        ryba)
                echo "ryba"
                ;;
        *)
                echo "cos innego"
                ;;
esac

# następny fragment działa dokładnie tak samo tyle że na instrukcji if
if [ "$COS" = "kot" -o "$COS" = "pies" ]; then
        echo "kot lub pies";
elif [ "$COS" = "ryba" ]; then
        echo "ryba"
else
        echo "cos innego"
```

fi

Sterowanie terminalem, czyli jak pisać kolorowo

Najprostszą metodą kolorowania napisów na terminalu (nie wymagającą żadnej biblioteki) jest bezpośrednie użycie sekwencji sterujących terminala. Główną wadą takiego rozwiązania jest to iż uzależniamy się od konkretnego typu terminala. Poniżej kilka przykładów:

```
echo -e "\\033[1mPOGRUBIENIE\\033[0m"
echo -e "\\033[31mCZERWONY\\033[0m"
echo -e "\\033[1;31mJASNO CZERWONY\\033[0m"
echo -e "\\033[34mGRANATOWY\\033[0m"
echo -e "\\033[37;40mSZARY NA CZARNYM TLE\\033[0m"
echo -e "\\033[1;33;40mZOLTY NA CZARNYM TLE\\033[0m"
```

Więcej na temat kolorowania w Kolorowe Powłoki[1044].

Możemy także decydować o miejscu wypisywania. Jeżeli chcemy tylko wypisywać w tej samej linii to wystarczy `echo -en "NAPIS DO WYPISANIA\r";`. Natomiast jeżeli chcemy pisać w dowolnym miejscu ekranu to potrzebna będzie nam odpowiednia instrukcja sterująca terminalem - np. `echo -e "\\033[10;30HNAPIS DO WYPISANIA"` spowoduje wypisanie od 30 znaku 10 linii.

Te same sekwencje sterujące terminalem (co najwyżej z dokładnością do znaków zabezpieczających) można użyć w programowaniu w C/C++.

Funkcje

Bash umożliwia również definiowanie funkcji, z których możemy potem korzystać jak z zwykłych poleceń:

```
#!/bin/bash

# deklaracja funkcji o nazwie wypisz, argumenty podstawiane są jako $1 $2 ...
wypisz() {
        echo "$1 World !!!"
}

# deklarację możnaby poprzedzić słowem kluczowym ”function” ale nie jest to wymagane,
# a na dodatek jest niekompatybilne z innymi sh

# wywolanie funkcji petla z jednym argumentem - Witaj
wypisz Witaj
```

Skrypty z opcjami

Przy zastosowaniu narzędzi takich jak `getopt`, `getopts` możliwe jest pisanie skryptów dobrze reagujących na przekazywane opcje i parametry (w taki sposób jak normalne unix’owe programy). Poniżej prezentuję rozwiązanie oparte na programie ”getopt”, pozwala ono na korzystanie zarówno z długich jak i krótkich opcji (łatwiejsza w zastosowaniu komenda getopts ma niestety problem z opcjami długimi). Alternatywne rozwiązanie oparte na awk zawarte jest w engine-base.sh.

```
#!/bin/bash

# przetworzenie argumentów określonych w wywołaniu getopt
# dalsze argumenty dostępne są w $1, $2, itd po zakończeniu pętli
# : -> poprzedzająca opcja wymaga argumentu
# :: -> poprzedzająca opcja może mieć argument
```

```
# uwaga o ile wymagane argumenty moga byc rozdzielane od opcji spacją o tyle opcjonalne nie
# (krótkie należy podawać zaraz po opcji, długie rozdzielając =)
eval set -- "`getopt -o xy:z:: -l opcja-x,opcja-y:,opcja-z:: -- "$@"`"
while true; do
 case $1 in
  -x|--opcja-x) echo "X";;
  -y|--opcja-y) echo "Y, argument \`$2'"; shift;;
  -z|--opcja-z) echo "Z, argument \`$2'"; shift;;
   # tuataj $2 może być pusty (gdy nie podano), ale jest zdefiniowany
  --) shift; break;;
 esac
 shift;
done
```

Linki i moje projekty

Zachęcam również do przyjrzenia się kilku prostym skryptom bashowym:

- arp_log.sh - bardzo prosty skrypcik logujący w zadanych odstępach czasu aktualny wygląd tablicy ARP[1045] do pliku
- fs_backup.sh - kolejny bardzo prosty skrypcik wykonujący backup wskazanej partycji na innej z wykorzystaniem rsync
- mysql_backup.sh - kolejny bardzo prosty skrypcik wykonujący backup baz danych MySQL
- progress_bar.sh - prosty skrypcik wypisujący kropki w trakcie działania innego procesu
- expr2pla.sh - prosty skrypt do generowania plików PLA (używanych np. przez espresso) na podstawie wyrażenia boolowskiego
- kamera.sh - robi zdjęcie aparatem cyfrowym aż się uda
- tv.vlc.sh - z wykorzystaniem DCOP uruchamia zestaw aplikacji do słuchania radia
- sshfs-via-tunel-mount.sh - z wykorzystaniem DCOP uruchamia termianle z zestawieniem tunelu ssh oraz montowaniem zasobu poprzez sshfs
- vhdl-build.sh - skrypt ułatwiający budowanie i uruchamianie testbench'ów VHDL
- Commons Config System - Skrypty przeznaczone do obsługi współdzielonego (sieciowo) konfiguracji pomiędzy kilkoma hostami. Korzysta do tego celu z sshfs oraz bzr i rsync, do startu i stopu korzysta z skryptów dodawanych do `.kde/env` i `.kde/shutdown` wyświetlających swój progressbar. Skrypt ponadto eksportuje klucze i certyfikaty szyfrujące oraz inne ustawienia przechowywane w $HOME oraz wykonuje "zaawansowane" tarowanie celów linków.
- rtv_recorder_and_transmiter - nagrywanie i przesylanie obrazu z karty TV z wykorzystaniem transcode, vlc, mplayer
- install_dom0.sh - skrypt automatyzujący instalacje systemu z użyciem debootstrapa i puppeta, warto zwrócić uwagę na rozwiązanie pętli z pytaniem o kontynuowanie
- beos_mail2mbox.sh - prosty skrypt bashowo-awkowy do konwersji skrzynki pocztowej z BeOSa do standardowego mbox'u
- net_stat.sh - prosty skrypt bashowo-awkowy do generowania statystyk transferu na wybranym interfejsie sieciowym
- mymenu_install.sh - skrypt instalujący menu w KDE (z ciekawszych rozwiązań warto wspomnieć o ustalaniu ścieżki katalogu w którym znajduje się plik skryptu, sprawdzaniu czy dwa katalogi nie są tym samym katalogiem, podmianie pierwszego wystąpienia napisu w pliku)
- engine-base.sh, engine-parse_in.sh, engine-parse_out.sh, engine-generate.sh, engine-config.sh, engine-edit_help.sh - dość rozbudowany skrypt bash'owo-awk'owy obsługujący moją stronę www
- System obsługi DIRopu
- System monitoringu pracy serwerów

Polecam także zajrzeć do artykułu poświęconego językowi AWK, który jest bardzo użyteczny w połączeniu z bash'em.

Więcej informacji o Bash'u: `man 1 bash`. Zobacz w Sieci: Advanced Bash-Scripting HOWTO[1046], Kurs BASHa[1047], Kurs basha[1048], Bourne Shell Scripting[1049].

Drobne skrypty shellowe

PROGRESS_BAR.SH

```
#!/bin/bash

# skrypt jest prostym rozwiązaniem wypisującym kropki podczas działania innego procesu
# (informujące o jego działaniu)

# urucahmiamy (w tle) jakiś proces o działaniu którego chemy informować
sleep 30 > /dev/null &

PID=`jobs -p 1`
while [ -e /proc/$PID -a "$PID" != "" ]; do
 # w zasadzie wystarczyłoby pobieranie PIDu powyższą metodą w pętli
 # i sprawdzanie czy niepusty,
 # jednak metoda z sprawdzaniem /proc wydaje się bardziej ogólna
 echo -n '.'
sleep 1
done
echo ""
```

KAMERA.SH

```
#!/bin/bash

# skrypt jest próbą zautomatyzowania wykonywania i pobierania zdjęcia z aparatu cyfrowego
# zabezpieczoną przed błędami transmisji (każde z poleceń wykonuje się do skutku)

if [ -f /tmp/kamera.sh_pracuje ]; then
 echo "Skrypt tworzacy fotki aktualnie dziala"
exit
fi

:> /tmp/kamera.sh_pracuje

echo " Przygotowywuje katalog roboczy"
katalog=/tmp/`date +%s`/
 mkdir $katalog
 cd $katalog

echo " Robie fotke"
while ( ! gphoto2 --capture-image ) do
  sleep 2;
 done

echo " Pobieram fotki"
while ( ! gphoto2 --get-all-files ) do
  sleep 2;
 done

echo " Kasuje fotki"
while ( ! gphoto2 --delete-all-files ) do
  sleep 2;
 done
```

```
rm -f /tmp/kamera.sh_pracuje
```

ADD_MD5_TO_PDF.SH

```
#!/bin/bash

# script to add md5 sum and other info to pdf for printing

STAMPFILE=`mktemp`
UP=false

INPUT="$1"
[ "$2" = "up" ] && UP=true

MD5=`md5sum "${INPUT}" | cut -f1 -d' '`
INFO=`pdfinfo "${INPUT}" | grep '^Page size:'`
INFO_X=`echo $INFO | awk '{print $3 * 10}'`
INFO_Y=`echo $INFO | awk '{print $5 * 10}'`
OFFSET_X="32"
if $UP; then
 OFFSET_Y=$[ $INFO_Y/10 - 64 ]
else
 OFFSET_Y="32"
fi

gs -o ${STAMPFILE} -sDEVICE=pdfwrite -g${INFO_X}x${INFO_Y} \
-c "/Helvetica-Bold findfont 8 scalefont setfont" \
-c "0 .8 .8 0 setcmykcolor" -c "${OFFSET_X} ${OFFSET_Y} moveto" \
-c "(MD5: ${MD5}  FILE: ${INPUT}) show" -c "showpage"

pdftk "${INPUT}" stamp ${STAMPFILE} output "${INPUT%.pdf}__MD5__.pdf"

\rm ${STAMPFILE}
```

ARP_LOG.SH

```
#!/bin/bash

while true; do
 date >> /var/log/arp.log
cat /proc/net/arp >> /var/log/arp.log
echo "" >> /var/log/arp.log
sleep 150
done
```

FS_BACKUP.SH

```
#!/bin/bash

CO=/
GDZIE=/dev/sdb1
TMP=/mnt/backup/local/root
```

```
mount $GDZIE $TMP
rsync -avx --delete --exclude="/DYSK.txt" $CO $TMP
echo "$GDZIE ($CO fs backup)" > $TMP/DYSK.txt
umount $TMP
```

MYSQL_BACKUP.SH

```
#!/bin/bash

savelog -m 640 -c 7 mysql.dump
mysqldump --defaults-extra-file="/etc/my.backu.acess.cfg" --all-databases > mysql.dump
# gdzie /root/.my.backu.acess.cfg:
#    [client]
#    host=localhost
#    user=backup
#    password=mysecret
#    #database=
# oczywiście plik ten powinien mieć odpowiednie prawa dostępu
```

NTPDATE_CRON.SH

```
#!/bin/bash

# zabezpieczenie przed zapetleniem spowodowanym zmiana czasu przez ntpdate
# i wywolywaniem skryptu w okreslonej godzinie przez cron'a
sleep 10

ntpdate-debian > /dev/null
```

BEOS_MAIL2MBOX.SH

```
#!/bin/bash

# skrypt konweruje skazany katalog z wiadomosciami w formacje skrzynki BeOS'owej
# na format standardowego mboxa (konwersja postaci linii From)
#
# katalog podawany jest jako pierwszy argument,
# listy zapisywane s do korespondencja_zbiorcza_nazwa_katalogu


if [ "$1" = "" ]; then
 echo "Musisz podać katalog"
exit
fi


for f in ./$1/*; do
 #echo "$f"; formail -x "Date:" < "$f"
 formail -a "Date:" < "$f" >> korespondencja_zbiorcza_tmp_$1
 echo "" >> korespondencja_zbiorcza_tmp_$1
done;
```

```
# Konwersja linii From do postaci : From user@host.domena    Thu Aug 19 19:43:33 2004
awk '
 # przypadek gdy mamy dzien tygodnia
 $1 == "From"  &&  $3 ~ "[A-Z][a-z][a-z]," {
  printf("%s %s %s %s %02d %s %s\n", $1, $2, substr($3,1,3), $5, $4, $7, $6);
 }

 # przypadek gdy nie mamy dnia tygodnia
 $1 == "From"  &&  $3 !~ "[A-Z][a-z][a-z]," {
  printf("%s %s %s %s %02d %s %s\n", $1, $2, "Xxx", $4, $3, $6, $5);
 }

 #pozostałe linie
 $1 != "From" {
  printf("%s\n", $0);
 }
' < korespondencja_zbiorcza_tmp_$1 > korespondencja_zbiorcza_$1
```

NET_STAT.SH

```
#!/bin/bash

# konfiguracja
RX_ERR="3072"
TX_ERR="3072"
INTERFACE=${1-eth0}
WAIT_TIME=0.1
FILENAME="/tmp/spped_$WAIT_TIME_$INTERFACE"

zczytaj() {
 cat /proc/net/dev /proc/uptime | tr ':' ' ' | awk '
  $1=="'$INTERFACE'" {
   printf("NEW_rec=%s; NEW_tran=%s; ", $2, $10)
  }
  END {
   printf("NEW_time=%s;\n", $1)
  }'
}

# inicjalizacja i wypisanie nagłówka
zczytaj > ${FILENAME}_new.txt
date +"%s,  %Y-%m-%d %H:%M:%S %z (%Z),  %A" --utc
echo -e "   RX    \t   TX"
sleep $WAIT_TIME

while true; do
 # zapamietujemy poprzednie jako stare
 sed 's%NEW%OLD%g'< ${FILENAME}_new.txt > ${FILENAME}_old.txt

# zczytujemy dane do pliku
 zczytaj > ${FILENAME}_new.txt

# wczytujemy pliki
  . ${FILENAME}_new.txt
 . ${FILENAME}_old.txt

# dokonujemy obliczeń
```

```
 TIME=`echo "scale=4; $NEW_time-$OLD_time" |bc`
RX_SPEED=`echo "scale=4; (8/1024)*($NEW_rec-$OLD_rec)/($TIME)" |bc`
TX_SPEED=`echo "scale=4; (8/1024)*($NEW_tran-$OLD_tran)/($TIME)" |bc`

# formatujemy wynik
 RX_SPEED=`env LANG=C printf "%09.4f" $RX_SPEED`
TX_SPEED=`env LANG=C printf "%09.4f" $TX_SPEED`

# wypisujemy wynik
 if [ `echo $RX_SPEED| cut -f1 -d.` -gt $RX_ERR -o `echo $TX_SPEED| cut -f1 -d.` -gt $TX_ERR ]
 ; then
   date +"%s,  %Y-%m-%d %H:%M:%S %z (%Z),  %A" --utc
   echo -en "$RX_SPEED\t$TX_SPEED\n"
else
   echo -en "$RX_SPEED\t$TX_SPEED\r"
fi

sleep $WAIT_TIME
done;
```

INSTALL_DOM0.SH

```
#!/bin/bash

if [ $# -ne 2 ]; then
 echo "USAGE $0 system_device hostname-instalowanego-dom0"
exit
fi

set -x
DEV=$1
MNT=/mnt/

parted $DEV "mklabel gpt"
parted $DEV "mkpart grub 0 1000kB";
parted $DEV "set 1 bios_grub on"
parted $DEV "mkpart raid 1MB 100%";

pvcreate ${DEV}2
vgcreate lvm0 ${DEV}2
lvcreate -L 12GB -n dom0 lvm0;
mkfs.xfs /dev/lvm0/dom0;

mount /dev/lvm0/dom0 $MNT

debootstrap stable $MNT ftp://ftp.icm.edu.pl/pub/Linux/debian/

echo "search vls.icm.edu.pl" > ${MNT}etc/resolv.conf;
echo "nameserver 2001:6a0:0:2000::18" >> ${MNT}etc/resolv.conf

echo "deb http://ftp.icm.edu.pl/pub/Linux/debian squeeze main non-free" > ${MNT}
etc/apt/sources.list;
chroot $MNT aptitude update
chroot $MNT aptitude install firmware-bnx2 firmware-bnx2x;

echo "deb http://ftp.icm.edu.pl/pub/Linux/debian squeeze main" > ${MNT}etc/apt/sources.list;
chroot $MNT aptitude update
chroot $MNT aptitude install debian-archive-keyring puppet grub;
```

```
mount -t proc proc /mnt/proc
mount -t sysfs sysfs /mnt/sys
mount -o rbind /dev/ /mnt/dev
grub-install --root-directory=${MNT} --recheck $DEV
set +x

answer="x"
while [ "$answer" != "n" ]; do

#echo "2001:6a0:0:21::d1:4 puppet.vls.icm.edu.pl" > ${MNT}etc/hosts
 chroot $MNT puppet agent --test --server puppet.vls.icm.edu.pl --fqdn $2.vls.icm.edu.pl

answer="x"
while [ "$answer" != "y" -a "$answer" != "n" ]; do
  echo "Ponowic pobieranie konfigu z puppet'a (y/n)?"
  read answer
done
done
:> ${MNT}etc/udev/rules.d/70-persistent-net.rules
```

SMTP.SH

```
#!/bin/bash

# wywołanie: while true; do netcat -l -p 25 -e smtp.sh; done

# wymagania: busybox z netcatem potrafiącym odpalić skrypt w -e
# na openwrt trzebabylo jako /bin/sh dać basha bo busybox chiał odpalać tylko komendy ↩
wbudowane

DOMAIN="opcode.eu.org"


IS_DATA=false; SENDER=""; RECEIVER="";
echo "220 $DOMAIN SMTP server by RRP `date -R`"

while read line; do
 echo "Q (data=$IS_DATA): Y $line" >> /tmp/smtp-log.txt

if $IS_DATA; then
  if echo "$line" | tr -d '\r' | awk '$0=="." {exit 0;} {exit 1}'; then
   IS_DATA=false
   echo -en "250 OK\r\n"
   continue
  fi

  ########################
  # WŁAŚCIWA TREŚĆ SKRYPTU #
  ########################
  # jest to powtarzane dla każdej linii przesylanej jako tresc maial (lacznie z naglowkami)
  # możemy sprawdzać nadawcę kopertowego: "$SENDER"
  # możemy sprawdzać adresata kopertowego: "$RECEIVER"
  # możemy sprawdzać aktualną linię z maila: "$line" tak jak to pokazano poniżej:

  echo "$line" | tr -d '\r' | awk '
   $1=="call" {
    cmd=sprintf("make_call.sh %s %s", $2, $3);
```

```
    }
    $1=="stop" {
     cmd=sprintf("killall .... ");
    }
    END {
     system(cmd);
    }
   ' > /dev/null 2> /dev/null &
   # UWAGA: musi to być wyciszone i przechodzić w tło

 else
   echo "$line" | tr -d '\r' | awk -v DOMAIN="$DOMAIN" '
   BEGIN {
    IGNORECASE=1
    FS="[ :\t]+"
   }
   $1 ~ "^(HE)|(EH)LO$" {
    printf("250 %s Hello %s, pleased to meet you\r\n", DOMAIN, $2);
    system( sprintf("echo \"R: 250 %s Hello %s, pleased to meet you\" >> /tmp/smtp-log.txt", ⤸
    DOMAIN, $2) );
    exit 0;
   }
   $1 ~ "^MAIL$" && $2 ~ "^FROM$" && $3 != "" {
    printf("250 OK\r\n");
    system( sprintf("echo \"R: 250 OK\" >> /tmp/smtp-log.txt") );
    exit 1;
   }
   $1 ~ "^RCPT$" && $2 ~ "^TO$" && $3 != "" {
    printf("250 OK\r\n");
    system( sprintf("echo \"R: 250 OK\" >> /tmp/smtp-log.txt") );
    exit 2;
   }
   $1 ~ "^DATA$" {
    printf("354 Enter message, ending with \".\" on a line by itself\r\n");
    system( sprintf("echo \"R: 354 Enter message, ending with \".\" on a line by itself\" >> ⤸
    /tmp/smtp-log.txt") );
    exit 3;
   }
   $1 ~ "^QUIT$" {
    printf("221 %s closing connection\r\n", DOMAIN);
    system( sprintf("echo \"R: 221 %s closing connection\" >> /tmp/smtp-log.txt", DOMAIN) );
    exit 4;
   }
   {
    printf("500 unrecognized command\r\n");
    system( sprintf("echo \"R: 500 unrecognized command (%s)\" >> /tmp/smtp-log.txt", $1) );
    exit 0;
   }
   '
   case $? in
    1) SENDER=`echo $line | tr -d '\r' | awk 'BEGIN {FS="[ :\t<>]+"} {print $3}'` ;;
    2) RECEIVER=`echo $line | tr -d '\r' | awk 'BEGIN {FS="[ :\t<>]+"} {print $3}'` ;;
    3) IS_DATA=true ;;
    4) exit ;;
   esac
  fi
 done;
```

SEND_SMS.SH

```
#!/bin/bash

##
## CONFIG
##

MODEMHOST="ip.ip.ip.ip"
MODEMPORT="port"

PORT="/tmp/vmodem.$MODEMHOST.$MODEMPORT"
NUMER_CSCA="+48501200777"
PIN=0000

function init_port() {
 socat pty,link=$PORT,b9600,echo=0 tcp:$MODEMHOST:$MODEMPORT &
 sleep 0.5
 #stty -F $PORT 115200 -echo parodd
}

# ON SOCAT REMOTE SIDE:
# screen -S socat sh -c "while true; do socat tcp-l:$MODEMPORT,reuseaddr,fork,bind=$MODEMHOST file:/dev/ttyUSB0; sleep 1; done"

# IN PROCMAILRC:
#:0 hw
#subject=|formail -xSubject:
#:0 h : /tmp/procmail_sending_sms
#|/etc/mail/procmail_users_scripts/send_sms.sh +$PHONE_NUMBER "$subject" 50



##
## INIT SCRIPT
##

echo "  -> Sending SMS \"$2\" to $1" >&2

if [ $# -lt 2 ]; then
        echo "USAGE: $0 numer_telefonu wiadomosc [num]"
echo "Optional \"num\" is uset to cut \"wiadomosc\" to max \"num\" characters"
        exit
fi

if [ $# -eq 3 ]; then
 msg=`echo $2 | cut -c 1-$3`
else
 msg=$2
fi

function recive_from_modem() {
 while read -t 1 l; do
  [ "$1" != "" ] && echo $1 >&2
 done < $PORT
}

function send_to_modem() {
 [ -e $PORT ] || init_port
echo -n -e "$1" > $PORT
```

```
 recive_from_modem
}



##
## SENDING MESSAGE
##

[ $PIN -ne 0 ] && send_to_modem "AT+CPIN=\"$PIN\"\015"
send_to_modem "ATV1Q0E1\015"
send_to_modem "AT+CSCA=\"$NUMER_CSCA\"\015"
send_to_modem "AT+CMGF=1\015"
send_to_modem "AT+CMGS=\"$1\"\015"
send_to_modem "$2\032"



##
## ENDING SCRIPT
##

sleep 10
recive_from_modem
echo "  -> SMS was send" >&2
```

SSHFS-VIA-TUNEL-MOUNT.SH

```
#!/bin/bash

# skrypt wymaga dostepu ssh do lokalnego hosta w oparciu o nazwę hosta zwracana przez ⤸
hostname
# oraz odpowiedniego skonfigurowania icm-ws_tunel w ~/.ssh/config (musi wskazywać na host ⤸
który montujemy)

if netstat -a -n --ip --inet6 | grep '5555.*LISTEN' > /dev/null; then exit; fi

konsole=$(dcopstart konsole-script)

session=$(dcop $konsole konsole currentSession)
dcop $konsole $session renameSession 'tunel'
dcop $konsole $session sendSession 'HISTFILE=""; export HISTFILE;'
dcop $konsole $session sendSession 'echo LACZENIE SIE Z TUNELEM: ssh sshfs-tunel'
dcop $konsole $session sendSession "ssh -t icm 'ssh -t sshfs-tunel \"ssh -R 5555:localhost:22 ⤸
`hostname`\"'"

session=$(dcop $konsole konsole newSession)
dcop $konsole $session renameSession 'sshfs'
dcop $konsole $session sendSession 'HISTFILE=""; export HISTFILE;'
dcop $konsole $session sendSession 'fusermount -u /home/rrp/SSHFS'
dcop $konsole $session sendSession 'sshfs -o nonempty -o workaround=rename sshfs-tunel:. ⤸
/home/rrp/SSHFS'
```

TUNEL_SSH_OVER_VLAN.SH

```
#!/bin/bash

# skrypt zestawia tunel ssh na połączeniu poprzest most WiFi
# most WiFi dostepny jest poprzez tagowane VLANy
# 172.16.16.0/28 - adresy w VLANie po naszej stronie
# 172.16.16.32/30 - adresy mostu WiFi
# 172.16.16.16/28 - adresy w VLANie po drugiej stronie

NUM=${1:-0}

# ID VLANu po naszej stronie
VLAN_ID=116
# nasze IP w VLANie (z maską w notacji CIDR)
VLAN_IP="10.10.1.9/28"
# IP Acess Pointa w naszym VLANie
VLAN_AP="10.10.1.1"

# IP sieci połączeniowej WiFi
WIFI_NET="10.10.1.32/30"

# IP sieci VLANu po drugiej stronie
REMOTE_NET="10.10.1.16/28"
# IP serwera tunelowego SSH
REMOTE_IP="10.10.1.30"
# dodatkowe opcje SSH
SSH_ARGS="-p4022"

# adresy tunelu SSH
TUN_LOCAL_IP="10.10.1.33"
TUN_REMOTE_IP="10.10.1.34"

# zestawienie VLANu do WIFI
modprobe 8021q
vconfig add eth0 $VLAN_ID
ifconfig eth0.$VLAN_ID   up  $VLAN_IP
ip route add $WIFI_NET   via $VLAN_AP
ip route add $REMOTE_NET via $VLAN_AP
ifconfig eth0.$VLAN_ID mtu 1492

# zestawienie Tunelu SSH
# wymagany wpis w sshd_config po stronie zdalnej (172.16.16.30):
#     PermitTunnel yes
screen ssh -w $NUM:$NUM root@$REMOTE_IP $SSH_ARGS -t \
'ifconfig tun'"$NUM"' up '"$TUN_REMOTE_IP"' netmask 255.255.255.252; bash'
# jakby logowac po kluczach to mozna by dodac screenowi -d -m
#       a tak to tgrzeba podac haslo i Ctrl+D ...
ifconfig tun$NUM up $TUN_LOCAL_IP netmask 255.255.255.252
ip route add 192.168.0.0/16 via $TUN_REMOTE_IP
ip route add default        via $TUN_REMOTE_IP
echo "nameserver 8.8.8.8" > /etc/resolv.conf

echo -e "NA SERWERZE SSH:\n\t"'echo "1" > /proc/sys/net/ipv4/conf/all/forwarding'
echo -e "NA ROUTERZE ZDALNEJ SIECI:\n\tip route add $TUN_LOCAL_IP/32 via LOCALNE.IP.SERWERA.
SSH;
 iptables -A POSTROUTING -t nat -o eth1 -s $TUN_LOCAL_IP -d 0.0.0.0/0 -p ! esp  -j SNAT --to-
 source IP.ZEWN.SERWERA.NAT"

ntpdate-debian
```

ALARM.SH

```
#!/bin/bash

SOUND="path/to/alarm/wav.file"

PATH="$PATH:/usr/bin"

aplay "$SOUND" 2> /dev/null

if [ `who | grep ${USER-""} | awk '$2 == ":0" {print "OK";}'` != "" ]; then
 export DISPLAY=:0.0
 ( kdialog --title "ALARM" --msgbox "INFO: $*" 2> /dev/null || echo "ALARM (kdialog error): ⤸
 $*" >&2 ) &
 sleep 5
 if [ `jobs -r | wc -l` -eq 1 ]; then
  aplay "$SOUND" 2> /dev/null
fi
else
 echo "ALARM (no X session): $*" >&2
fi
```

ZEGAR.SH

```
#!/bin/bash

function print_time() {
 echo -en "`date -d @$TIMESTAMP +"%A, %Y-%m-%d %H:%M:%S  %Z (%z)"`\r"
echo -en "`date -d @$TIMESTAMP --utc +"%A, %Y-%m-%d %H:%M:%S  %Z (%z)"`\r"
echo "UNIX time: $TIMESTAMP seconds since the epoch"
}

# wyrównanie wyświetlania
TIMESTAMP_tmp=`date +%s`
TIMESTAMP=$TIMESTAMP_tmp
while [ $TIMESTAMP_tmp -eq $TIMESTAMP ]; do
 TIMESTAMP_tmp=$TIMESTAMP
 TIMESTAMP=`date +%s`
done

for f in `seq 1 5`; do
 print_time | aosd_cat -l 3 -p 0 -x 20 -y 20 -B black -b 35 -n "courier bold 16" -d 3 -f 100  ⤸
 -u 800 -o 100
 TIMESTAMP=$(( $TIMESTAMP + 1 ))
done
```

Programowanie w AWK

Tytułem wstępu zaznaczę że GAWK jest wersją interpretatora języka skryptowego AWK[1050], opracowywaną w ramach projektu GNU, posiadającym własne rozszerzenia (zwane rozszerzeniami GNU). Z faktu iż jest to język skryptowy wynika iż program wykonywany jest tożsamy z kodem źródłowym - nie ma etapu kompilacji.

Składnia języka jest dość podobna do C, oczywiście nie znajdziemy tu zdecydowanej większości funkcji z biblioteki standardowej C oraz są też bardzo istotne różnice w samej budowie programu. Podstawową z tych różnic jest to iż program w języku AWK składa się z bloków ”warunek { instrukcje }” oraz (opcjonalnie) definicji funkcji ”function nazwa (argumenty) { instrukcje }”. Taka budowa składni wynika z głównego zastosowania do jakiego został stworzony AWK - przetwarzania tekstu.

Skrypty AWK mogą być zapisywane go pliku - wywołujemy przez `gawk -f sciezka_pliku`, mogą też być podawane w linii poleceń: `gawk 'tresc programu'` (dla bezpieczeństwa treść tak podawanego programu należy ujmować w ” - inaczej powłoka może z nią zrobić coś niedobrego interpretując zawarte tam znaki specjalne … . Możemy też w pliku z poleceniami AWK dodać jako pierwszą linię `#!/usr/bin/gawk -f` oraz dodać mu argument wykonywalności - będzie zachowywał się jak zwykły program przyjmujący dane z stdin lub plików podanych jako argumenty.

Tym razem zaczniemy tradycyjnie … od ”Witaj Świecie” (zakładam że plik w którym umieszczane są przykładowe programy nazywa się ”hello.awk” i ma atrybut wykonywalności):

```
#!/usr/bin/gawk -f

{ printf "Hello World !!!\n" }
```

Wykonanie ./hello.awk spowoduje oczekiwanie na dane wejściowe z sdtin … po wpisaniu każdej kolejnej linii program wypisze ”Hello World !!!” (aby zkończyć Ctrl+D) … natomiast ./hello.awk jakis_plik wypisze ileś razy ten napis i zakończy działanie … ale chyba nie o to chodziło. Spróbujmy wiec tak:

```
#!/usr/bin/gawk -f

BEGIN { printf "Hello World !!!\n" }
```

Teraz ./hello.awk działa jak byśmy oczekiwali (identycznie też ./hello.awk jakis_plik) - wypisuje ”Hello World !!!” i kończy.

Powyższe przykłady obrazują (chyba) podstawową własność języka AWK to co jest zawarte w klamrach wykonywane jest dla każdej pasującej linii (przed klamrami może stać warunek w stylu `FNR == 1`, co oznacza numer_rekordu == 1 bądź wyrażenie regularne np. `^T[ye]` co oznacza że linia zaczyna się od T po którym jest y lub e) oraz są specjalne identyfikatory - poznany BEGIN i END wykonywane odpowiednio na początku i końcu programu. Widzimy też funkcję printf - bliźniaczo podobną do tej z C (powyżej pominąłem nawiasy ale jest to to samo co `printf("Hello World !!!\n")`).

Widzimy tez że nie ma średników, gdy instrukcja kończy się z końcem linii (ale nic nie stoi na przeszkodzie aby je stawiać), są natomiast konieczne gdy w jednej linii mamy kilka instrukcji.

Kolejnym ważnym aspektem są zmienne i tablice. Nie ma potrzeby określania typu zmiennej, wszystkie zmienne poza zmiennymi wewnętrznymi funkcji są globalne. AWK używa tablic asocjacyjnych - indeksem w tablicy może być cokolwiek (liczba lub napis), nie ma potrzeby ustalania długości tablicy ani typu elementów w niej przechowywanych; poniżej przykład ilustrujący tablice w AWK. Wszystkie te cechy ilustruje poniższy przykładowy kod

```
#!/usr/bin/gawk -f

function moja_funkcja( zmienna1 ) {
	zmienna2 = sin (2);
	zmienna1 = sprintf("%s @ %f", zmienna1, zmienna2)
	return zmienna1;
}

BEGIN {
	tablica["Hello"]="World";
	tablica["Witaj"]="Świecie";
	for (INDEX in tablica)
		printf("%s %s !!!\n", INDEX, tablica[INDEX])
}

FNR == 1 {
	zmienna3 = "Ala ma kota"
```

```
        moja_funkcja(zmienna3);
        # to jest komentarz
        # uwaga: miedzy nazwą funkcja a otwierającym nawiasem nie może być białych ⤸
        znaków
        printf("%s %s\n", zmienna3, zmienna1)
}
FNR == 2 {
        zmienna4 = 5
        zmienna4 = moja_funkcja(zmienna4);
        printf("%s %s\n", zmienna3, zmienna4)
}
```

Warto zaznaczyć że dość często wykorzystuje się tablice w których interesuje nas tylko index - jest tak dlatego że gwarantuje to unikalność (nie może być dwóch takich samych indeksów), zatem gdy zbieramy napisy i chcemy wyeliminować powtórzenia przyjemnym rozwiązaniem jest `nasza_tablica[napis]=0`. Oczywiście mogą być też tablice dwu lub więcej wymiarowe: `nasza_tablica[indeks1,index2]=wartosc`. Na koniec warto zaznaczyć że oprócz powyższego wywołania for mamy też zupełnie standardowe pętle for, while znane z C.

Innym częstym zagadnieniem jest wypisanie wszystkich pól od wskazanego do końca rekordu. Możemy do tego celu użyć funkcji uzytkownika jak w poniższym przykładzie:

```
function to_end(i,  tmp) {tmp=$i; for (++i; i<=NF; i++) tmp = tmp OFS $i; return tmp;}
{
        print to_end(3)
}
```

Więcej (w szczególności jak budować wyrażenia regularne oraz wykaz predefiniowanych funkcji) - patrz `man 1 gawk` oraz:

- Ściąga do AWK[1051] - kilkustronicowa broszurka z najważniejszymi komendami
- Opis języka AWK[1052] - kilkudziesięciostronnicowy podręcznik do AWK
- Effective AWK Programming[1053] - podręcznik rozprowadzany wraz z pakietem GAWK
- AWK @ Wikibooks[1054]
- An Awk Primer[1055]

Skrypty AWK'owe często wykorzystywane są w skryptach powłoki (bash, ...), nie stoi też nic na przeszkodzie aby wykorzystywać je w C - np. przez system() lub exec(); . Warto tutaj zwrócić uwage na przekazywanie zmiennych bashowych do awk - mozna to zrobić (w przypadku prostych skryptów poprzez umieszczenie skryptu w podwójnych cudzysłowiach i zabezpiecznu tych znaków specjalnych których bash nie powinien dotykać poprze \ - np. `awk "(\$1==$bash) {print \$0}"` (zmienną bashową którą przekazujemy awk jest `$bash`) albo poprzez przekazanie zmiennej opcją -v - np. `awk -v temp=$temp '($1==temp) {print $0}'`.

AWK jest także przydatnym narzędziem do wykonywania obliczeń (obliczania średnich, odchyleń, różnic, etc) na danych będących wynikami jakiś symulacji lub pomiarów.

Linki i moje projekty

Zamieszczam tutaj krótki skrypcik który usuwa zadany katalog ze zmiennej systemowej PATH - `PATH=`echo $PATH | awk 'BEGIN {RS=":"; FS="\n"} $1!="/sciezka/katalogu/do/usuniecia" {printf("%s:",$1)}'``, a także przykłady skryptów bashowych wykorzystujących AWK:

- beos_mail2mbox.sh - konwersja skrzynki pocztowej z BeOSa do standardowego mbox'u
- net_stat.sh - statystyki transferu na zadanym interfejsie sieciowym
- smtp.sh - prosty serwer SMTP (wykorzystujący do nasłuchu netcat'a), umożliwia sterowanie (wykonywanie zdalnych poleceń przy pomocy odpowiednio spreparowanych maili (jest wymagającym w teorii tylko busybox'a zastępnikiem jakiegoś serwera, pocztowego który otrzymane maile przekazuje do skryptu wyciągającego z nich potrzebne dane przy pomocy grep/cut/awk)
- engine-parse_menu.awk, engine-generate.sh - dość rozbudowany skrypt bash'owo-awk'owy obsługującego moją stronę www (więcej strona projektu)

Python

Python[1056] jest interpretowanym językiem skryptowym o dość ciekawych możliwościach. Najistotniejszą różnicą w stosunku co do innych języków jest wyznaczanie bloków kodu wyłącznie przy pomocy wcięć. Język ten posiada zaawansowaną obsługę list (m.in. operatory wykonujące działanie na każdym z elementów listy). Poniżej przedstawiam przykład prezentujący podstawowe cechy tego języka, zachęcam także do zapoznania się z podręcznikiem Pythona w Wikibooks[1057], How to Think Like a Computer Scientist: Learning with Python[1058], Non-Programmer's Tutorial for Python 3[1059], Non-Programmer's Tutorial for Python 2.6[1060], Think Python[1061], A Beginner's Python Tutorial[1062] oraz skryptem do mirrorngu mediawiki i skryptem do zapisywania załączników z poczty[1063].

Sporą zaletą tego języka jest całkiem dobra integracja z C++ z użyciem "Python C API" lub "Boost Python". Możliwe jest zarówno stworzenie biblioteki w C++, którą wykorzystamy w kodzie pythonowym jak też (trochę bardziej złożone w realizacji) wywołanie kodu pythonowego z C++. Zamieszczam dwa przykłady związane z współpracą C++ i Pythona:

- mojmodul.cpp i mojmodul_test.py - moduł w C++ wykożytywany w Pythonie
- run_from_cpp.cpp i run_from_cpp_script.py - skrypt pythonowy wykożytywany w C++ (w tym wystawienie obiektu z C++ do Pythona, zarówno przez argument, jak i jako obiektu globalnego)

Poniżej krótki przykład programu w pythonie. Ponaddto zamieszczam skrypt do sprawdzania czy podany adres ip zawiera się w którejś apachowej regułce "allow from" (i wskazania tej reguły) - jest to przydatne przy szukaniu błędów w rozbudowanych plikach tego typu.

```
#!/usr/bin/python
# coding: utf-8

u'''Jest to forma komentarza wielolinijkowego,
jeżeli wsytępuje on na początku bloku funkcji czy całego skryptu
pełni on funkcję dokumentacyjną
poprzedzenie otwierających 3 apostrofów literą u pozwala na zapis zanaków unikodu'''

# na początek standardowe Hello World
print "Hello World"

# teraz deklaracja funkcji
def info(napis, numer=10):
 """to również jest komentaż wielolinijkowy
 warto zwórcić uwagę iż blok instrukcji składających się na deklarację funkcji
 wydzielony jest WYŁĄCZNIE przy pomocy wcięc"""

# jak widać pętla for przypomina trochę tą z basha
 for i in range(numer+1):
  print napis
  if (i % 2 == 0):
   print "^^^ to był parzysty wypis (i=%d)" % i
  else:
   print ""


# a to znowu kod główny
info("Ala ma kota", 3)

print "=============="

# obsługa list ...
lista_liczb = [1, 3, 6, 3]
print "YY " .join(["XX: %d\n" % n for n in lista_liczb])

lista_napisow = ['aaa', 'bbb', 'ccc', 'ddd']
print lista_napisow[1]
print lista_napisow[-1]
```

```
print "elemnt ccc ma indeks %d" % lista_napisow.index("ccc")
```

TEST_IP.PY

```
#!/usr/bin/python
# coding: utf-8

import sys
import re
from netaddr import *
import socket
import argparse

def apache_to_verbose(abrev_cidr):
    """
    Funkcja bazująca na cidr_abbrev_to_verbose z python-netaddr służąca parsowaniu ⤸
    apachowego niepełnego zapisu sieci IP
    """
    start = ''
    tokens = []
    prefix = 32

    if abbrev_cidr == '':
        return abbrev_cidr
    if ':' in abbrev_cidr:
        return abbrev_cidr
    if '/' in abbrev_cidr:
        return abbrev_cidr

    if '.' in abbrev_cidr:
        tokens = abbrev_cidr.split('.')
    else:
        tokens = [abbrev_cidr]

    if 1 <= len(tokens) <= 4:
        for i in range(4 - len(tokens)):
            tokens.append('0')
            prefix=prefix-8
    else:
        #    Not a recognisable format.
        return abbrev_cidr

    if prefix is None:
        try:
            prefix = nprefix
        except ValueError:
            return abbrev_cidr

    return "%s%s/%s" % (start, '.'.join(tokens), prefix)


def test_ip(net, ip, verbose=False, net_txt=None):
    """
    Funkcja sprawdzajaca czy podany zakres adresów ip (ip) zawiera się w badanym zakresie ⤸
    adresów IP (net)

    :param net:     sieć w której szukamy podanego adresu ip / podsieci (string)
```

```
    :param ip:      szukany adres ip / podsieci (IPSet)
    :param verbose: (opcjonalny) decyduje o tym czy wypisujemy zawsze czy tylgo gdy zawiera ⤸
    się
    """
    ipnet = IPNetwork(apache_to_verbose(net))
    ipset = IPSet([ ipnet ])
    r = ipset.issuperset(ip)
    if r or verbose:
        if net_txt is None:
            net_txt = net
        print "{} ({}) {}" .format(str(ipnet), net_txt, r)


def check_and_test_ip(net, ip, verbose=False, resolve_dns=True):
    # if entry is ip address
    if re.match("^[0-9:./]*$", net):
            test_ip(net, ip, verbose)
    # otherwise try resolve as DNS name
    else:
        try:
            dns_addrs = socket.getaddrinfo(net, 0);
            for dns_addr in dns_addrs:
                test_ip(dns_addr[4][0], ip, verbose, net)
        except:
            pass



# parser agumentów linii poleceń
# do argumentów można się też dostawać bezpośrednio poprzez sys.argv:
#     len(sys.argv) - ilość elementów tej tablicy
#     str(sys.argv[0]) - napis odpowiadający nazwie wywołania programuz
#     str(sys.argv[1]) - napis odpowiadający pierwszemu argumentoi progrumu

parser = argparse.ArgumentParser(description='Looking for a matching network (from input file ⤸
or STDIN, \
                                              using apache network notation for allow/deny ⤸
                                              rules) for the specified IP')
parser.add_argument('IP', help='search IP address')
parser.add_argument('-a', "--apache", action="store_true",
                    help='use apache config file format as input (parse only "allow from" ⤸
                    rules')
parser.add_argument('-n', "--numeric_only", action="store_true", help='don\'t resolve DNS ⤸
from input')
parser.add_argument('-v', "--verbose", action="store_true", help='print all input entry')
parser.add_argument('-i', "--input", action="store", metavar='FILE', help='input file (if not ⤸
specified STDIN)')
args = parser.parse_args()


# parsowanie szukanego adresu IP do formy IPSet
testipnet = IPNetwork( str(args.IP) )
testipset = IPSet([ testipnet ])


# ustalenie źródła z którego czytane są sieci
if args.input:
    print "Find for {} ({}) in {}" .format(str(testipnet), str(args.IP), str(args.input))
    nets_input = open(str(args.input), 'r')
else:
```

```
    print "Find for {} ({}) in {}" .format(str(testipnet), str(args.IP), "STDIN")
    nets_input = sys.stdin


# ustalenie formatu źródła sieci
if args.apache:
     for line in nets_input:
        if re.match("^allow\s*from", line, re.IGNORECASE):
            # podział linii na listę słów przy pomocy dowolnego białego znaku
            # poprzez line.split(None,2) możemy podzielić na 2 pierwszych separatorach
            # poprzez line.split("SEP") możemy określić separator
            # poprzez re.split("REGEXP", line) możemy dzieć wg wyrażenia regularnego
            nets = line.split();

            # usuwamy z listy dwa pierwsze argumentu (lista nadal numerowana jest od zera)
            del nets[1], nets[0];

            # przetwarzamy każdy element pozostały na liście słów
            for net in nets:
                check_and_test_ip(net, testipset, args.verbose, args.numeric_only)
else:
    for line in nets_input:
        check_and_test_ip(line.strip(), testipset, args.verbose, args.numeric_only)
```

MOJMODUL.CPP

```
// kompilacja: g++ -shared -fPIC -lpython2.7 -lboost_python -I/usr/include/python2.7/ ⤸
mojmodul.cpp -o mojmodul.so

#include <iostream>
#include <string>

/// przykłądowe funkcje i klasa, które będa udostępnione jako moduł pythonowy
std::string funkcja1() {
 std::cout << "Ala ma kota\n";
 return "Kot ma Alę";
}

int funkcja2(int a) {
 return a + 10;
}

class Klasa {
 public:
  Klasa(const std::string& val) {
   this->val = val;
  }
  std::string show() const {
   return "Val is: " + val;
  }
 private:
  std::string val;
};

std::string funkcja3(const Klasa& w) {
 return w.show() + " call from funkcja3";
}
```

```
/// definicja modułu pythonowego

#include <boost/python.hpp>
using namespace boost::python;

BOOST_PYTHON_MODULE(mojmodul) {
 def("funkcja1", funkcja1);
 def("f2", funkcja2);

class_<Klasa>("Klasa", init<std::string>())
  .def("show", &Klasa::show)
  .def("f3", funkcja3)
 ;

def("funkcja3", funkcja3);
}
```

MOJMODUL_TEST.PY

```
#!/usr/bin/python

import sys
sys.path.append('/path/to/dir/with/mojmodul.so/')

from mojmodul import *

print "XXX"
print "from python: " + funkcja1()

num=4
print f2(num) * 2

k = Klasa('AAAA')
print k.show()

print funkcja3(k)
print k.f3()
```

RUN_FROM_CPP.CPP

```
// kompilacja: g++ -lpython2.7 -lboost_python -I/usr/include/python2.7/ run_from_cpp.cpp

#include <boost/python.hpp>

namespace engine {
 class Testuj {
 public:
  Testuj() {
   std::cout << "inicjalizuje Testuj" << "\n";
   liczba=0;
   liczba2=0;
  }
```

```
  void setLiczba(int a) {
   std::cout << "ustawiam liczba na: " << a << "\n";
   liczba = a;
  }

  void run(int b) {
   std::cout << "Wynik wynosi: " << liczba + b << "\n";
  }

  int liczba2;
 private:
  int liczba;
 };

BOOST_PYTHON_MODULE(EngineScripts) {
  boost::python::class_<Testuj>("Testuj", boost::python::init<>())
   .def("run", &Testuj::run)
   .def_readwrite("value",    &Testuj::liczba2)
  ;
 }
}

int main(int, char **) {
 engine::Testuj *tt = new engine::Testuj();
 tt->setLiczba(3);

engine::Testuj *ttt = new engine::Testuj();
 ttt->setLiczba(4);

// initialise python
 Py_Initialize();

try {
  // initialise and import EngineScripts module
  //   through the calling initEngineScripts() we not need build python module separately via:
  //   g++ -shared -fPIC -lpython2.7 -lboost_python -I/usr/include/python2.7/   EngineScripts.⤸
  cpp -o EngineScripts.so
  engine::initEngineScripts();
  PyRun_SimpleString("from EngineScripts import *" );

  // prepare script to run
  boost::python::object main = boost::python::import("__main__");
  boost::python::object global(main.attr("__dict__"));

  // export object to python
  global["CPPobject"] = boost::python::ptr(ttt);

  boost::python::object result = boost::python::exec_file("./run_from_cpp_script.py", global, ⤸
  global);
  boost::python::object script = global["foo"];

  if(!script.is_none()) {
    // run scripts
    std::cout << "run\n";
    std::cout << "return = " << boost::python::extract<int>(  script(boost::python::ptr(tt)) )⤸
    << "/n";
    std::cout << "finish\n";
  }

  global["abc"] = boost::python::ptr(tt); // podobnie możemy dostawać się do ⤸
```

```
   zmiennych/obiektów tworzonych w pythonie
   PyRun_SimpleString("abc.run(20)" );
  } catch(boost::python::error_already_set &) {
   PyErr_Print();
   exit(-1);
  }

 std::cout << "ttt->liczba2 = " << ttt->liczba2 << std::endl;

 //Py_Finalize();
  return 0;
 }
```

RUN_FROM_CPP_SCRIPT.PY

```
def foo(ott):
 ott.run(4)
 CPPobject.run(0)
 CPPobject.value = 1587
 return 2193
```

MEDIAWIKI_MIRROR.PY

```
#!/usr/bin/python3

# skrypt tworzy mirror podanych linków artykułów mediawiki z użyciem api.php,
# wczytuje obecne w nich linki do bazy celem umożliwienia rekurencyjnego mirroringu

import sys, os, re, time

from http.client     import HTTPSConnection
from urllib.parse    import quote, unquote
from urllib.request import urlretrieve
import json
import sqlite3

dbConn = None
httpHeaders = {'User-Agent':'MediawikiMirrorBot'}

def doQuery(propFull, propShort, pageID, title, conn,  extraInfo = None, baseSubProp = None, ⤸
subProps = None):
 queryStr = '/w/api.php?action=query&continue&format=json&titles=' + quote(title)
 if subProps != None:
  queryStr += '&' + propShort + 'prop=' + subProps
queryStr += '&prop=' + propFull + '&' + propShort + 'limit=100'

contTxt = ''
contFlag = True
 retInfoBase = []
 retInfoExtra = None
 while contFlag:
  conn.request('GET', queryStr + contTxt, headers=httpHeaders)
  res = conn.getresponse()
  if res.status != 200:
```

```
      raise IOError('GET HTTP code: ' + r1.status + r1.reason + ' for ' + propFull + ' request: ' + queryStr)
    data = json.loads(res.read().decode())

    if pageID == None:
      tmp = list(data['query']['pages'].keys())
      if len(tmp) != 1:
        raise ValueError('Multiple pages in reply for: ' + queryStr)
      else:
        pageID = tmp[0]

    if data['query']['pages'][pageID]['title'] != title:
      raise ValueError('Invalid title in reply for: ' + queryStr)
    if 'continue' in data:
      contTxt = '&' + propShort + 'continue=' + quote(data['continue'][propShort + 'continue'])
    else:
      contFlag = False

    if propFull in data['query']['pages'][pageID]:
      if baseSubProp == None:
        retInfoBase += data['query']['pages'][pageID][propFull]
      else:
        for tmp in data['query']['pages'][pageID][propFull]:
          retInfoBase += [ tmp[baseSubProp] ]
    if (not contFlag) and extraInfo != None and extraInfo in data['query']['pages'][pageID]:
      retInfoExtra  = data['query']['pages'][pageID][extraInfo]

return {'base': retInfoBase, 'extra': retInfoExtra}


def getDstDir(title, create=False):
  dstDir = ''
for d in title.split('/'):
    dstDir += d + '/'
    if not os.path.isdir(dstDir):
      if create:
        os.mkdir(dstDir)
      else:
        return None
  return dstDir


def getArticle(title, site, getLinks=True, save=True, forceRefresh=False):
  saveDir = getDstDir(site + '/' + title)
  if saveDir != None and not forceRefresh:
    return { 'meta': {'site': site, 'title': title}, 'dir': saveDir, 'status': ['done'] }

conn     = HTTPSConnection(site)

# get page ID and page text
  queryStr = '/w/api.php?action=query&continue&format=json&titles=' + quote(title) \
        + '&prop=revisions&rvprop=content&rvlimit=1&redirects=1'
conn.request('GET', queryStr, headers=httpHeaders)
  res = conn.getresponse()
  if res.status != 200:
    raise IOError('GET HTTP code: ' + r1.status + r1.reason + ' for revisions request: ' + queryStr)
  data = json.loads(res.read().decode())

if '-1' in data['query']['pages'] and 'missing' in data['query']['pages']['-1']:
```

```
    raise ValueError('Page Missing')

  txt    = ''
  pageID = 0
   status  = []
   for pID in data['query']['pages']:
    if data['query']['pages'][pID]['revisions'][0]['contentformat'] != 'text/x-wiki' and \
       data['query']['pages'][pID]['revisions'][0]['contentformat'] != 'text/plain':
      raise ValueError('Invalid contentformat in reply for: ' + queryStr)
    if data['query']['pages'][pID]['revisions'][0]['contentmodel'] != 'wikitext' and \
       data['query']['pages'][pID]['revisions'][0]['contentmodel'] != 'Scribunto':
      raise ValueError('Invalid contentmodel in reply for: ' + queryStr)
    if txt != '':
     raise ValueError('More one page in reply for: ' + queryStr)
    if data['query']['pages'][pID]['title'] != title:
     oldTitle = title
     if 'normalized' in data['query'] and \
         data['query']['normalized'][0]['to'] == data['query']['pages'][pID]['title']:
       title = data['query']['normalized'][0]['to']
       status += ['normalized']
     if 'redirects' in data['query'] and \
         data['query']['redirects'][0]['to'] == data['query']['pages'][pID]['title']:
       title = data['query']['redirects'][0]['to']
       status += ['redirect']

     if 'normalized' in data['query'] and data['query']['normalized'][0]['from'] != oldTitle ⤸
     and \
        'redirects' in data['query'] and data['query']['redirects'][0]['from'] != oldTitle:
       status = []

     if status != []:
      saveDir = getDstDir(site + '/' + title)
      if saveDir != None and not forceRefresh:
       return { 'meta': {'site': site, 'title': title}, 'dir': saveDir, 'status': status + [⤸
       'done'] }
     else:
      raise ValueError('Invalid title in reply for: ' + queryStr)

    txt = data['query']['pages'][pID]['revisions'][0]['*']
    pageID = pID

  # get page authors
   res = doQuery('contributors', 'pc', pageID, title, conn,  extraInfo='anoncontributors')
   contributors = {'users': res['base'], 'anonymous': res['extra']}

  # get redirect to this article
   res = doQuery('redirects', 'rd', pageID, title, conn,  baseSubProp='title')
   redirects = res['base']

  if getLinks:
    # get page level links (images, templates)
    res = doQuery('images', 'im', pageID, title, conn,  baseSubProp='title')
    images = res['base']

    res = doQuery('templates', 'tl', pageID, title, conn,  baseSubProp='title')
    templates = res['base']

    # get page sub level links (links, interwiki, ...)
    res = doQuery('links', 'pl', pageID, title, conn,  baseSubProp='title')
    links = res['base']
```

```
  res = doQuery('iwlinks', 'iw', pageID, title, conn)
  for tmp in res['base']:
   links += [ tmp['prefix'] + ':' + tmp['*'] ]

  res = doQuery('langlinks', 'll', pageID, title, conn)
  for tmp in res['base']:
   if tmp['lang'] == 'en' or tmp['lang'] == 'pl':
    links += [ tmp['lang'] + ':' + tmp['*'] ]
 else:
  images = None
  templates = None
  links = None

# save meta, links and data(text)
 if save:
  saveDir = getDstDir(site + '/' + title, create=True)

  infoFile = open(saveDir + '/info.txt', 'w')
  infoFile.write(json.dumps(
   {'site': site, 'title': title, 'pageID': pageID, 'contributors': contributors, \
    'images': images, 'templates': templates, 'links': links},
   indent=1,
   ensure_ascii=False)
  )
  infoFile.close()

  infoFile = open(saveDir + '/data.txt', 'w')
  infoFile.write(txt)
  infoFile.close()
 else:
  saveDir = None

conn.close()
 return { 'meta': {'site': site, 'title': title, 'pageID': pageID, 'contributors': ⤸
 contributors}, \
          'txt': txt, 'dir': saveDir, 'redirects': redirects, 'status': status, \
          'images': images, 'templates': templates, 'links': links }


def getImage(title, site, save=True):
 conn     = HTTPSConnection(site)

res = doQuery('imageinfo', 'ii', None, title, conn,  subProps='user|userid|url|comment')

# save meta and data (download image file)
 if save:
  dstDir = getDstDir(site + '/' + title, create=True)
  infoFile = open(dstDir + '/info.txt', 'w')
  infoFile.write(json.dumps(res['base'], indent=1, ensure_ascii=False))
  infoFile.close()

  urlretrieve( res['base'][0]['descriptionurl'], dstDir + '/info.html' )
  urlretrieve( res['base'][0]['url'], dstDir + '/data.' + res['base'][0]['url'].rsplit('.', 1)⤸
  [1] )
                                                    # '/' + re.sub(' ', '_', unquote(res['base'][0][⤸
                                                    'url'].rsplit('/', 1)[1]))
 conn.close()
 return {'dir': dstDir, 'title': title, 'site': site, 'info': res['base']}
```

```
def parseLink(link, base = None, level = 0, site = None):
 if site != None:
  ss = site.split('.', 3)
  lang=ss[0]
  project=ss[1]
 else:
  lang='pl'
  project='wikipedia'

langs={'pl', 'en', 'de', 'es', 'fr', 'it'}
 projects={ 'w':'wikipedia', 'b':'wikibooks', 'c':'commons.wikimedia', 'wikt':'wiktionary', ⤸
 's':'wikisource', \
            'q':'wikiquote', 'species':'wikispecies', 'voy':'wikivoyage', 'v':'wikiversity', ⤸
            \
            'wikipedia':'wikipedia', 'wikibooks':'wikibooks', 'commons':'commons.wikimedia', ⤸
            \
            'wiktionary':'wiktionary', 'wikisource':'wikisource', 'wikiquote':'wikiquote', \
            'wikispecies':'wikispecies', 'wikivoyage':'wikivoyage', 'wikiversity':⤸
            'wikiversity' }
 title=''

ls = link.split('#', 2)[0].split(':', 3)
 titleStartAt = 2

for i in range(0, len(ls)-1):
  tmp = ls[i]
  if tmp in langs:
   lang = tmp
  elif tmp in projects:
   project=projects[tmp]
  else:
   titleStartAt = i
   break

for i in range(titleStartAt, len(ls)-1):
  title += ls[i] + ':'
title += ls[len(ls)-1]

newLevel = level + 10
 if base != None and title.startswith(base):
  subtitle = title.replace(base, '', 1)
  if subtitle.startswith("#"):
   return None
  elif subtitle.startswith("/") or subtitle.startswith(":"):
   newLevel = level + 1

return { 'site': lang + '.' + project + '.org', 'title': title, 'level': newLevel }



def saveFullArticle(site, title, level, verify):
 aInfo = getArticle(title, site)

if not os.path.isfile(aInfo['dir'] + '/info.txt'):
  aInfo = getArticle(title, site, forceRefresh=True)

if not 'done' in aInfo['status']:
  # mirror all images from article
```

```
for tmp in aInfo['images']:
 res = chechInDatabase(aInfo['meta']['site'], tmp, level, 0)
 if res['status'] != 'done':
  tmpInfo = getImage(tmp, aInfo['meta']['site'])
  addToDatabase(tmpInfo['site'], tmpInfo['title'], tmpInfo['dir'], res['level'], res[⤸
  'verify'])

# mirror all templates from article
for tmp in aInfo['templates']:
 res = chechInDatabase(aInfo['meta']['site'], tmp, level, 0)
 if res['status'] != 'done':
  try:
   tmpInfo = getArticle(tmp, aInfo['meta']['site'], getLinks=False)
   addToDatabase(tmpInfo['meta']['site'], tmpInfo['meta']['title'], tmpInfo['dir'], res[⤸
   'level'], res['verify'])
  except ValueError as e:
   print("\n[" + tmp + "] ERROR:", e)
else:
 infoFile = open(aInfo['dir'] + '/info.txt', 'r')
 data = json.load(infoFile)
 aInfo['links'] = data['links']

# add all links from article to mirroring queue
for tmp in aInfo['links']:
 tmpInfo = parseLink(tmp, title, level, site)
 if tmpInfo == None:
  continue

 res = chechInDatabase(tmpInfo['site'], tmpInfo['title'], tmpInfo['level'], 0)
 if res['status'] != 'done':
  addToDatabase(tmpInfo['site'], tmpInfo['title'], None, res['level'], res['verify'])
 else:
  if tmpInfo['level'] < res['level']:
   addToDatabase(tmpInfo['site'], tmpInfo['title'], None, tmpInfo['level'], res['verify'])

# update database:
dbCursor = dbConn.cursor()
#   - when change title delete old entry
if title != aInfo['meta']['title']:
 dbCursor.execute("INSERT INTO normalized VALUES (?, ?, ?)", (title, aInfo['meta']['title'], ⤸
 aInfo['meta']['site']))
 dbCursor.execute("DELETE FROM pages WHERE site=? AND title=?", (site, title))
#   - add article and all redirects to him to database
dbCursor.execute("INSERT INTO pages VALUES (?, ?, ?, ?, ?)", (aInfo['meta']['site'], aInfo[⤸
'meta']['title'], aInfo['dir'], level, verify))
if 'redirects' in aInfo:
 for rt in aInfo['redirects']:
  dbCursor.execute("INSERT INTO pages VALUES (?, ?, ?, ?, ?)", (aInfo['meta']['site'], rt, ⤸
  aInfo['dir'], level, 2))
dbConn.commit()



def connecteDatabase():
 global dbConn
dbConn=sqlite3.connect('wikiPages.db')


def chechInDatabase(site, title, level = 999, verify = 0):
```

```
    dbCursor = dbConn.cursor()
    dbCursor.execute("SELECT path, level, verify FROM pages WHERE site = ? AND title = ?", (site,⤸
    title))
    res = dbCursor.fetchone();
    if res is None:
        # not in mirroring queue
        return { 'status': 'new', 'level': level, 'verify': verify }
    elif res[0] is None:
        # in mirroring queue or mirrored
        if level > res[1]:
            level = res[1]
        return { 'status': 'waiting', 'level': level, 'verify': res[2] }
    else:
        # mirroring is done
        return { 'status': 'done', 'level': res[1], 'verify': res[2] }


def addToDatabase(site, title, directory, level, verify):
    dbCursor = dbConn.cursor()
    dbCursor.execute("INSERT INTO pages VALUES (?, ?, ?, ?, ?)", (site, title, directory, level, ⤸
    verify))
    dbConn.commit()


def createDatabase():
    global dbConn

    if os.path.isfile("wikiPages.db"):
        if not dbConn is None:
            dbConn.close()
        os.rename("wikiPages.db", "wikiPages.db.old")

    dbConn=sqlite3.connect('wikiPages.db')
    dbCursor = dbConn.cursor()
    dbCursor.execute("CREATE TABLE pages (site TEXT, title TEXT, path TEXT, level INT, verify ⤸
    INT DEFAULT 0, PRIMARY KEY (site, title) ON CONFLICT REPLACE)")
    dbCursor.execute("CREATE TABLE normalized (oldTitle TEXT, newTitle TEXT, site TEXT)")
    dbConn.commit()


def loadLinksToDatabase():
    dbCursor = dbConn.cursor()

    f = open('entry_points-ignore.txt', 'r')
    for line in f:
        line = line.strip()
        if line != '':
            tmpInfo = parseLink(line)
            dbCursor.execute("SELECT * FROM pages WHERE site = ? AND title = ?", (tmpInfo['site'], ⤸
            tmpInfo['title']))
            if dbCursor.fetchone() is None:
                dbCursor.execute("INSERT INTO pages VALUES (?, ?, '__IGNORE__', -1, 1)", (tmpInfo['site'],⤸
                tmpInfo['title']))
    dbConn.commit()
    f.close()

    f = open('entry_points-auto.txt', 'r')
    for line in f:
        line = line.strip()
        if line != '':
```

```
            tmpInfo = parseLink(line)
            dbCursor.execute("SELECT * FROM pages WHERE site = ? AND title = ?", (tmpInfo['site'], ⤸
            tmpInfo['title']))
            if dbCursor.fetchone() is None:
                dbCursor.execute("INSERT INTO pages VALUES (?, ?, NULL, 1, 1)", (tmpInfo['site'], tmpInfo[⤸
                'title']))
    dbConn.commit()
    f.close()

f = open('entry_points-manual.txt', 'r')
    for line in f:
        line = line.strip()
        if line != '':
            tmpInfo = parseLink(line)
            dbCursor.execute("SELECT * FROM pages WHERE site = ? AND title = ?", (tmpInfo['site'], ⤸
            tmpInfo['title']))
            if dbCursor.fetchone() is None:
                dbCursor.execute("INSERT INTO pages VALUES (?, ?, NULL, 123, 1)", (tmpInfo['site'], ⤸
                tmpInfo['title']))
    dbConn.commit()
    f.close()


def downloadLevel(level):
    dbCursor = dbConn.cursor()

for row in dbCursor.execute('SELECT * FROM pages WHERE level = ' + str(level) + ' AND path ⤸
    IS NULL AND verify=1'):
        info1 = 'Download [' + row[0] + "] " + row[1]
        sys.stdout.write(info1)
        sys.stdout.flush()
        t = time.time()
        msg=None

        try:
            saveFullArticle(row[0], row[1], row[3], row[4])
            t = time.time() - t
            info2 = 'done [' + str(round(t)) + 's]'
        except Exception as e:
            t = time.time() - t
            if e.args[0] == 'Page Missing':
                dbCursor2 = dbConn.cursor()
                dbCursor2.execute("INSERT INTO pages VALUES (?, ?, ?, ?, ?)", (row[0], row[1], ⤸
                '__MISSING__', row[3], row[4]))
                dbConn.commit()
                info2 = 'NOT FOUND [' + str(round(t)) + 's]'
            else:
                print("\n        ERROR:", e)
                dbCursor2 = dbConn.cursor()
                dbCursor2.execute("INSERT INTO pages VALUES (?, ?, ?, ?, ?)", (row[0], row[1], ⤸
                '__ERROR__', row[3], row[4]))
                dbConn.commit()
                info2 = 'ERROR [' + str(round(t)) + 's]'

        sys.stdout.write(' ' * (128 - len(info1) - len(info2)) + info2 + '\n')

        if t < 2:
            t = 2
        time.sleep(1.4 * t)
```

```
def printLinksToFix():
 if os.path.isfile('fixWikiLink.sh'):
  raise IOError('fixWikiLink.sh exist')
 f = open('fixWikiLink.sh', 'w')

dbCursor = dbConn.cursor()
 sql1='SELECT DISTINCT n.site, n.oldTitle, n.newTitle FROM normalized AS n JOIN pages AS p'
sql2='ON (n.site=p.site AND n.newTitle=p.title AND p.level=1)'
for row in dbCursor.execute(sql1 + ' ' + sql2):
  if row[0] == "pl.wikipedia.org":
   print('fixWiki pl "' + row[1] + '" "' + row[2] + '"', file=f)
  elif row[0] == 'en.wikipedia.org':
   print('fixWiki en "' + row[1] + '" "' + row[2] + '"', file=f)
  elif row[0] == 'pl.wikibooks.org':
   print('fixWiki pl "b:' + row[1] + '" "b:' + row[2] + '"', file=f)
  elif row[0] == 'en.wikibooks.org':
   print('fixWiki en "b:' + row[1] + '" "b:' + row[2] + '"', file=f)

f.close()
```

Programowanie w PHP

PHP[1064] jest językiem skryptowym o składni podobnej do C pomyślanym głównie o zastosowaniach w dynamicznym tworzeniu stron WWW (w zasadzie taka mieszanka C, AWK i Bash'a z dodatkiem sporej gamy funkcji webowych). Pomimo iż skrypty CGI można tworzyć w każdym z języków - konieczne jest tylko wysłanie jako pierwszej linii nagłówka Content-type (np. "Content-type: text/html"), drugiej linii pustej a potem mamy już treść to właśnie głównie przez te funkcje oraz dobrą integrację z HTML PHP jest do tego chyba najwygodniejszy.

Podobnie jak w AWK nie musimy definiować typu zmiennej, ale możemy uczynić to tak samo jak robiliśmy w C, pętle oraz instrukcje wyboru wyglądają identycznie jak w C, również operatory matematyczne, logiczne, porównania są takie same. Funkcje deklarujemy jak w C. Możemy korzystać z komentarzy C (/* do */), C++ (// do końca linii) oraz shela (# do końca linii). Podobnie jak w AWK mamy tutaj tablice asocjacyjne (oczywiście tablice możemy też indeksować klasycznie liczbami ...):

```
<?php
// od takiego znacznika zaczyna się kod PHP to co poza nimi traktowane jest jako dane ...
// kiedyś często pisało się tylko <? ale obecnie jest to mocno niezalecane
// - mamy przecież też <?xml ...>
 $nasza_tablica["Ala"]="ma kota";
 $nasza_tablica["Tom"]="ma psa";

reset ($nasza_tablica);// - ustawienie aktualnej pozycji na początek tablicy
 // w tym wypadku nie jest to niezbędne,
 // ale gdy robimy dwa kolejne przeglądania z użyciem each jest to niezbędne
 while ( list($klucz, $wartosc) = each($nasza_tablica) ) {
  echo "nasza_tablica[$klucz] = $wartosc<br />\n";
  // do wypisywania służy echo, printf, ...
 }

// a to bardzo wygodna funkcja do debugowania/testowania
 // - rekurencyjnie wyświetla zawartośc tablicy
 print_r($nasza_tablica);
?>
```

Do ważnych instrukcji należy zaliczyć `include` oraz `include_once`, które włączają podany po nich plik (ta druga sprawdza czy nie był już włączony i włącza tylko gdy nie był - wygodne przy włączaniu plików z definicjami funkcji). Mamy tez możliwości programowania obiektowego.

Reszta programowania i (jak sądzę) siła tego języka tkwi w bogactwie funkcji - do obsługi napisów, protokołów

sieciowych (np. header()), plików (np. fopen(), itd.), baz danych, połączeń sieciowych (np. fsockopen()), … . Więcej w dokumentacji PHP[1065].

Na koniec warto zaznaczyć że dziedziny zastosowań PHP nie ograniczają sie tylko do WWW - może służyć do pisania innego rodzaju skryptów i aplikacji. W szczególności skrypty php możemy wykonywać jak każdy inny skrypt - wykonywalny plik tekstowy z `#!/usr/bin/php4` (uwagi: kod dalej musi być w <?php i ?> oraz wymaga odpowiedniej paczki PHP - np. w Debianie "php4-cli").

Linki i moje projekty

Zachęcam także do zapoznania się z moim projektem związanym z programowaniem w PHP: prosty system do publikacji news'ów, logowanie do phpmyadmin w oparciu o konto shellowe i plik /.my.cnf, elementem silnika tej strony opartym o php, system informacji o stawianiu serwerów gry sieciowej, System obsługi DIRopu (część webowa).

Zobacz w Sieci: PHP - tutoriale i artykuły php, manual, php, skrypty[1066], PHP @ Wikibooks[1067], PHP and MySQL Programming[1068].

Podstawy technologii baz danych

Modele danych[1069] są zbiorami zasad dotyczących: definicji danych, operowania danymi oraz integralności danych. Wyróżnić możemy hierarchiczny, sieciowy, relacyjny i obiektowy model danych. Oprócz nich mamy często też doczynienia z prostymi modelami danych (np. plik tekstowy o ustalonej strukturze jednak bez odniesień do zawartości innych plików).

W modelu hierarchicznym mamy doczynienia z związkami pomiędzy rekordami (nadrzędność i podrzędność), różnymi typami rekordów, rekordy posiadają klucze (unikalne identyfikatory). Dane w tym modelu mają strukturę drzewiastą, utworzoną przez rekordy nadrzędne i podrzędne, nie da się usunąć rekordu bez usuwania rekordów podrzędnych wobec niego. Przykładem takiej bazy danych jest system plików. W modelu sieciowym oprócz zwykłych pól posiadamy pola kolekcji zawierające odniesienia do innych rekordów.

W modelu relacyjnym[1070] (RDBMS[1071]) relacje zazwyczaj reprezentowane są przez tabele stanowiące nieuporządkowany zbiór rekordów[1072] o atrybutach określonych w kolumnach. Kolumny, również stanowiące nieuporządkowany zbiór, określają typy wartości pól[1073]. Pola są wartościami atomowymi lub wartością NULL. Wyróżnia się klucze główne (stanowiące unikalny identyfikator rekordu w tabeli) oraz klucze obce stanowiące powiązania pomiędzy tabelami - kolumny będące kluczami obcymi muszą mieć ten sam typ co klucz główny tabeli do której mają prowadzić klucze obce. Powiązania takie tworzy się przez wstawianie do kolumny klucza obcego odpowiednich wartości klucza głównego. Operacje na danych opisuje algebra relacyjna[1074] możliwa jest:

- selekcja
- rzut - wybór podzbioru wierszy
- iloczyn kartezjański - w sensie teorii mnogości, w wyniku otrzymujemy iloczyn ilości wierszy i sumę ilości kolumn
- równo-złączenie - z iloczynu kartezjańskiego wybieramy tylko wiersze spełniające równość pomiędzy dwiema kolumnami o tym samym typie z dwóch tabel (klucz główny = klucz obcy)
- złączenie naturalne - z równo-złączenia usuwamy powtórzenia kolumn
- złączenie zewnętrzne - podobnie jak złączenie naturalne tyle że gdy nie ma pasującego to wstawiamy wiersz z tabeli uprzywilejowanej a w miejscu pól z drugie wstawiamy NULL
- suma - w sensie teorii mnogości
- przecięcie - w sensie teorii mnogości
- różnica - w sensie teorii mnogości

Baza zapewnia także integralność danych, w szczególności integralność referencyjną - dotyczącą kluczy głównych i obcych (nie jest możliwe usunięcie wiersza na który wskazują klucze obce bez usuwania wierszy wskazujących, nie jest możliwe dodanie wiersza z kluczem obcym wskazującym na nieistniejący wiersz, nie jest możliwe dodanie wiersza powodujące zduplikowanie klucza głównego. Z zachowaniem poprawnością danych w modelu relacyjnym związane są więzy. Są to narzucone przez projektanta ograniczenia i dodatkowe relacje które muszą być spełniane przez wartości danego pola.

W modelu obiektowym występuje niejednorodna struktura danych zawierająca (obok wartości - atrybutów) także procedury. W modelu tym występuje "jednoznaczna tożsamość", oznacza to że mamy te same atrybuty i meto-

dy, ale różne obiekty (danego typu). Procedury związane związane z obiektem nazywane są metodami. Wzorce wg których tworzone są obiekty (typ obiektu) określany jest mianem klasy. Model ten umożliwia dziedziczenie, czyli tworzenie klas potomnych na bazie już istniejących przez ich modyfikację oraz zapewnia enkapsulację czyli ukrycie szczegółów implementacji tak aby nie były dostępne z zewnątrz ("to co w środku nieważne").

Drugą sprawą z teorii baz danych, którą trzeba poruszyć jest normalizacja danych[1075] i tzw. postać normalna[1076]. Ma to na celu unikanie powtórzeń, redukcję anomalii w modyfikacji danych oraz uproszczenie reguł integralności. Anomaliami nazywamy efekt uboczny modyfikacji, wstawiania lub usuwania danych pociągający za sobą zbyt duże zmiany. Wyróżnia się kilka postaci normalnych, każda kolejna jest tą bliższą ideałowi:

— 1NF - każdy atrybut niekluczowy jest funkcyjnie zależny od klucza głównego ("bez powtórzeń ...")
— 2NF - dodatkowo każdy atrybut niekluczowy jest w pełni zależny od klucza głównego ("... pola zależą od klucza ...")
— 3NF - dodatkowo każdy atrybut niekluczowy jest bezpośrednio zależny od klucza głównego ("... od całego klucza ...")

SQL

SQL[1077] jest językiem zapytań, wykorzystywanym do komunikacji z bazą danych, może być używany w trybie interaktywnym, lub poprzez zapytania wykonywane przez aplikację na potrzeby jej funkcjonowania. Istnieje wiele implementacji systemów bazodanowych korzystających z SQL, praktycznie każdy z nich posiada jakieś własne rozszerzenia tego standardu, lub nie implementuje pewnych jego fragmentów. Do pracy interaktywnej na ogół wykorzystuje się oprogramowanie opracowane specjalnie dla danej bazy danych i często dostarczane wraz z nią, natomiast do przesyłania zapytań z aplikacji wykorzystać można biblioteki dedykowane danemu serwerowi bazodanowemu (np. C API MySQL lub wbudowane biblioteki php dla baz danych) lub sterownik standardu ODBC[1078].

SQL posiada polecenia umożliwiające tworzenie, modyfikowanie i usuwanie baz danych oraz tabel, jak również wstawiania, kasowania pobierania i modyfikowania danych, umożliwia także nakładanie ograniczeń (więzów) na wprowadzane dane. SQL nie rozróżnia wielkości liter w nazwach poleceń.

```
    -- utworzenie bazy danych
    CREATE DATABASE `TESTOWA`;
    -- zmiana nazwy bazy danych
    ALTER DATABASE `TESTOWA` RENAME TO `test-db`;
    -- wybranie jej jako domyślnej, inaczej do tabeli aaa trzeba by się odwoływać
↪ przez test-db.aaa
    USE `test-db`;
    -- tworzymy tabele uzytkownicy
    --  z określonym kluczem głównym na kolumnie UID, która jest automatycznie
↪ zwiększana
    --  indeksem na kolumnie name (która może mieć wartość nieokreśloną)
    --  unikalnością na kolumnie PESEL (która nie może być pusta)
    CREATE TABLE `uzytkownicy` (
            `UID` INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY ,
            `name` VARCHAR( 25 ) NULL ,
            `PESEL` BIGINT( 11 ) UNSIGNED NOT NULL ,
            INDEX ( `name` ) ,
            UNIQUE ( `PESEL` )
    );
    -- dodaliśmy kolumnę password typu tekstowego
    -- o zmiennej długości wynoszącej maksymalnie 30 znaków
    --  została ona wstawiona po kolumnie name
    ALTER TABLE `uzytkownicy` ADD `password` VARCHAR( 30 ) NOT NULL AFTER `name`;
    -- zmieniliśmy definicję kolumny name tak aby napisy porównywane były wg
↪ kodowania utf8
    --  z uwzględnieniem kolejności języka polskiego
    ALTER TABLE `uzytkownicy` CHANGE `name` `name` VARCHAR( 25 )
            CHARACTER SET utf8 COLLATE utf8_polish_ci NULL DEFAULT NULL;
    -- wymagamy aby uzytkownicy byli urodzeni przed 1992 rokiem
    --  (ze względu na budowę peselu i prostotę tego warunku dopuszczamy urodzonych
↪ po 1999 ...)
```

```
ALTER TABLE `uzytkownicy` ADD CONSTRAINT `wiek` CHECK(PESEL<92000000000);
-- tworzymy kolejna tabelkę - na dane teleadresowe
CREATE TABLE `dane` (`UID` INT UNSIGNED NOT NULL,
        `type` TINYINT UNSIGNED NULL, `value` VARCHAR(50));
-- dodajemy wiąz unikalności określający iż cała
--  kombinacja identyfikatora użytkownika typu i wartości pola ma być unikalna
↪  ...
ALTER TABLE `dane` ADD CONSTRAINT `unikalnosc` UNIQUE(`UID`, `type`, `value`);
-- bawimy się wstawianiem i modyfikacją danych
INSERT INTO `uzytkownicy` VALUES (NULL, NULL, NULL, 91000000000);
INSERT INTO `uzytkownicy` VALUES (NULL, NULL, "super tajne", 91000000000);
INSERT INTO `dane` (`UID`, `value`) VALUES (0, NULL);
INSERT INTO `uzytkownicy` VALUES (NULL, 'rrp', "super tajne", 10000000000);
-- korzystamy z podzapytania przy wstawianiu
INSERT INTO `dane` VALUES (
        (SELECT UID FROM `uzytkownicy` WHERE name='rrp'),
        1, 'rrp@localhost' );
-- podzapytań możemy używać także w warunkach oraz w części zwracającej wyniki
SELECT name FROM `uzytkownicy` WHERE UID=(SELECT max(UID) FROM `uzytkownicy`);
-- robimy join'a (połączenie kilku tabelek), korzystając z aliasów
SELECT u.UID, u.name, d.value FROM `uzytkownicy` AS u
        JOIN `dane` AS d ON u.UID=d.UID;
SELECT u.UID, u.name, d.value FROM `uzytkownicy` AS u
        LEFT JOIN `dane` AS d ON u.UID=d.UID;
SELECT COUNT(*) AS 'ilosc' FROM `dane`;
SELECT COUNT(*) AS 'ilosc' FROM `dane` WHERE value IS NOT NULL;
```

(uwaga poniższe były testowane tylko w **PostgresSQL**, ale myślę że conajwyżej po niewielkich zmianach powinny zadziałać w MySQL)

```
-- jeżeli przed dodaniem warunku unikalności były dane z nim sprzeczne, możemy je
↪  łatwo wyszukać:
SELECT UID, type, value FROM dane GROUP BY UID,type,value HAVING count(UID)>1;
-- dodajemy komentarz do kolumny
COMMENT ON COLUMN `uzytkownicy`.PESEL IS 'w przyszłości zrobić kontrolę sumy';
-- gdybyśmy zapomnieli o primary key i indeksie przy tworzeniu tabeli to możemy
↪  dadać do istniejącej:
--  ALTER TABLE uzytkownicy ADD INDEX(name);
--  ALTER TABLE uzytkownicy ADD PRIMARY KEY (UID);
-- możemy także usuwać kolumny, więzy itd:
--  ALTER TABLE uzytkownicy DROP COLUMN "name";
-- dodawanie do tabeli klucza obcego, z kasowaniem kaskadowym
--  usunięcie użytkownika z `uzytkownicy` spowoduje usunięcie jego wpisów z
↪  `dane`
--   jakby było bez "ON DELETE CASCADE" to proba usunięcia gdy są wpisy w `dane`
--   zakonczyłaby się błędem
ALTER TABLE dane ADD FOREIGN KEY (UID) REFERENCES uzytkownicy(UID) ON DELETE
↪  CASCADE;
```

(uwaga poniższe były testowane tylko w **MySQL**)

```
-- danie użytkownika TESTER z hałem ABCDEF i pełnymi prawami do bazy TESTOWA
--  w PostgresSQL komenda createuser (więcej w )
--  np, createuser -s -d -U postgres USERNAME utworzy uzytkownika z pełnią praw
CREATE USER 'TESTER'@'localhost' IDENTIFIED BY 'ABCDEF';
GRANT USAGE ON * . * TO 'TESTER'@'localhost' IDENTIFIED BY 'ABCDEF' WITH \
        MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0 MAX_UPDATES_PER_HOUR 0
↪  MAX_USER_CONNECTIONS 0 ;
GRANT ALL PRIVILEGES ON `TESTOWA` . * TO 'TESTER'@'localhost';
--  w PostgresSQL użytkownik ma domyślnie pełnie praw do bazy o takiej samej
↪  nazwie ...
--  dodatkowe uprawnienia można przyznawać analogicznie do powyższego kodu - np.
```

```
--   GRANT ALL PRIVILEGES ON tabela TO user;
--  wyświetlenie uprawnień \dp tabela
-- zmiana hasła użytkownika
UPDATE USER SET password=password('GHIJKL') WHERE user='AAAAAA';
-- zmiana swojego hasła
SET password = password('jakieshaselko');
--  w PostgresSQL: alter user "nazwa_uzytkownika" with password 'jakieshaselko';
-- przeładowanie uprawnień (wymagane aby działało nowe hasło)
FLUSH PRIVILEGES;
-- usunięcie użytkownika i przyznanych mu praw
DROP USER 'AAAAAA'@'localhost';
-- wyświetlenie baz danych
--  w PostgresSQL polecenie \l w psql
SHOW DATABASES;
-- wyświetlenie tabel
--  w PostgresSQL polecenie \d w psql
--- w SQLlite  .tables
SHOW TABLES;
-- pokazanie struktury tabeli w SQLlite: .schema tabela
-- wylosowanie zbioru rekordów (losowa kolejność 5 pierwszych)
SELECT * FROM `uzytkownicy` ORDER BY rand() LIMIT 5;
```

SQL umożliwia ponadto deklarowanie zmiennych, funkcji i procedur działających po stronie serwera, tworzenie widoków ("wirtualnych" tabel opartych na zapytaniu o tabele istniejące), uruchamianie zaplanowanych działań w momencie wykonywania określonej operacji na bazie danych (trigger), korzystanie z transakcji (grup poleceń, które np. w wypadku niepowodzenia któregoś z nich mogą zostać cofnięte i są wykonywane w sposób niepodzielny). Powyższe przykłady zostały przygotowane dla MySQL 5. Więcej informacji można uzyskać korzystając np. z komendy `help` w konsoli wywołanej poleceniem `mysql` - np. `help CREATE TRIGGER`, `help START TRANSACTION`. Warto wspomnieć także o `mysqldump` zrzucającym wybraną (lub z opcja `--all-databases` wszystkie) bazę danych w postaci tekstowego SQL na standardowe wyjście oraz o `pg_dump -cif nazawa_plikubazadanych` pełniącym analogiczną funkcję w PostgresSQL.

Zobacz też: dokumentacja składni SQL dla MySQL[1079], MySQL @ Wikibooks[1080].

XML, XSL ITP

W zasadzie XML[1081] nie jest ani językiem programowania, ani nawet rzeczywistym formatem danych, jest to jedynie składnia służąca do tworzenia formatów danych. Jednak XSL[1082] spełnia już wszystkie wymogi bycia językiem programowania. Na XSL składają się mechanizmy takie jak:

- XSLT[1083] (XSL Transformations) - opis przetwarzania dokumentu XMLowego
- XPath[1084] (XML Path Language) - sposób odwołań do elementów dokumentu XML
- XQuery[1085] (XML Query Language) - język zapytań

Prezentuje dwa dokumenty XSL wykorzystywane przez silnik tego serwisu - base.xsl, softdb.xsl. Pierwszy z nich odpowiedzialny jest za zamianę źródeł dokumentów w XHTML lub LaTeX (posiada dwa tryby, w trybie LaTeX musi być wykonywany przez parser XSL2.0 ze względu na użycie fn:replace). Drugi odpowiedzialny jest za generowanie XML'a używanego w źródłach strony z bazy oprogramowania/komend.

Przy debugowaniu kodów XMLowych niekiedy przydaje się skrypt do odnajdywania miejsca otwarcia niezamkniętego znacznika (niekiedy - np. gdy dopuszczane jest zagnieżdżanie danego znacznika - parsery wskazują jako miejsce błędu koniec pliku). Prezentuje jedno-linijkowiec ułatwiający takie wyszukiwanie (przykład dla znacznika <p>)

```
tr '\n' ' ' < body.xml | sed -e 's#</p>#\n#g' | grep '.*<p>.*<p>'
```

Bardzo przydatnym narzędziem do przetwarzania plików xml jest xmlstarlet[1086] umożliwiający np. łatwe wydobycie wartości jakiegoś tagu lub jego atrybutu. Dla przykładu jeżeli chcemy pobrać wartość atrybutu deadline z wszystkich znaczników TASK możemy posłużyć się następującym plikiem xsl (i wywołać go komendą `xalan -xsl plik.xsl -in plik.xml`):

```
<?xml version="1.0"?>
<xsl:stylesheet version="2.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
>
        <xsl:output method='text' encoding='utf-8' />
        <xsl:strip-space elements="*" />
        <xsl:template match="TASK">
                <xsl:value-of select="@deadline"/>
        </xsl:template>
        <xsl:template match="*|@*|processing-instruction()">
                <xsl:apply-templates select="*|@*|processing-instruction()" />
        </xsl:template>
</xsl:stylesheet>
```

Albo możemy po prostu użyć komendy `xmlstarlet sel -t -v '//*/TASK/@deadline' plik.xml`
Zobacz w Sieci: XML & XSL[1087], XML - Managing Data Exchange[1088]

VHDL

VHDL[1089] jest jednym z dwóch (obok Verilog[1090]) najpopularniejszych języków opis sprzętu[1091]. Służy do opisu (na różnych poziomach abstrakcji) cyfrowego układu elektronicznego, celem symulacji jego działania oraz automatycznej syntezy rzeczywistego układu (w postaci schematu złożonego z układów dyskretnych, projektu układu scalonego, czy też wsadu do układów programowalnych FPGA[1092] itp). Najbardziej abstrakcyjne poziomy opisu mają problemy z syntezowalnością, dlatego jeżeli myślimy o wykorzystaniu VHDL do syntezy najbezpieczniej jest operować poziomem przesłań między-rejestrowych (Register transfer level[1093]). Ponadto należy się stosować do pewnych szablonów tak aby narzędzia syntezy bezproblemowo interpretowały kod. Istnieją także HDL dla układów analogowych oraz kilka innych języków HDL dla układów cyfrowych (np. AHDL[1094]).

Trochę innym podejściem jest SystemC[1095], będący w istocie biblioteką dla C++ umożliwiającą opisywanie (i symulowanie tak zbudowanych systemów) sprzętu w ramach kodu C++. Kod taki może być automatycznie konwertowany na VHDL i z niego może być dokonywana synteza.

- vhdl-build.sh - skrypt ułatwiający budowanie i uruchamianie testbench'ów VHDL
- moj_pakiet.vhdl - tworzenie własnego pakietu, deklaracje funkcji i procedur, konwersja pomiędzy liczbami całkowitymi a wektorami bitów
- multiplexer.vhdl - deklaracja prostego komponentu na przykładzie multipleksera, korzystanie z parametrów ogólnych, kierunki portów
- multiplexer_3to1.vhdl - wykorzystanie utworzonej jednostki w tworzeniu innej, zaprezentowanie rożnych instrukcji wyboru (innych pomysłów na multiplekser)
- nor.vhdl - operacje logiczne, procesy z wait, zmienne, operacje matematyczne, pętle, …
- przerzutnik.vhdl - podstawowy element pamiętający (wspomniane także o tym jak tworzy się zatrzaski)
- rejestr_przesuwny.vhdl - budowa rejestru przesuwnego w oparciu o proces oraz w oparciu o łączenie przerzutników, pokazuje wykorzystanie generate-for oraz generate-if, wybieranie podzestawu bitów z wektora bitowego
- automat.vhdl - przykład jawnej konstrukcji automatu
- tester.vhdl - przykład pisania testbench'ów, uwagi na temat syntezowalności, używanie literałów, wygodne wstawianie komponentu

Zobacz w Sieci: Plansze do wykładów z UCYF[1096], Verilog HDL quick reference card[1097].

VHDL-BUILD.SH

```
#!/bin/bash

SRC_DIR=$PWD
DIR=`mktemp -d`
cd $DIR

TO_RUN="${1-testbench}"
```

```
exit_on_error() {
 echo " failed"
echo ""
echo -e "\\033[1;33;40m\tERROR MESAGE:\\033[0m"
echo -e "\\033[1;31m$error\\033[0m" | sed -e 's#^\(.*\)$#\t\t\1#g'
echo ""
#echo ”We don’t remove temporary files in $DIR”
 #echo ” use contents of this directory to debug”
 #echo ” and press enter to remove it”
 #read f
 #rm -fr $DIR
 #exit $1;
 return $1
}

build_and_run() {
 # najpierw stosujemy preprocesor C (dzięki temu możemy uzywać np. #include)
 echo "C-style preprocesing and importing to GHDL work-library:"
for f in $SRC_DIR/*.vhdl; do
  ff=`basename $f`
  echo -ne "\t* $ff ..."
  gcc -x c -P -E $f 2>/dev/null | sed '1,2d' > $ff
  # FIXME o co chodzi z tymi 2 dodatkowymi liniami ...

  ghdl -i $ff 2>/dev/null
  echo " done"
done

# potem dopiero analizujemy (”kompilujemy”)
 is_error=0
 echo "VHDL analyze:"
for f in *.vhdl; do
  echo -ne "\t* $f ..."
  # a dopiero potem wykonyujemy analize vhdl
  if ! error=`ghdl -a $f 2>&1`; then
   exit_on_error -2
   is_error=$?
  else
   echo " done"
  fi
 done
 if [ $is_error -ne 0 ]; then
  return $is_error
 fi

# a na koniec elaboracja i ...
 echo -n "VHDL elaborate $TO_RUN ..."
if ! error=`ghdl -m $TO_RUN 2>&1`; then
  exit_on_error -3
  return $?
else
  echo " done"
fi

# uruchomienie i wygenerowanie przebiegu do podanego pliku
 # jest dużo różnych opcji odpalania powstałego pliku wykonywalnego
 echo -e "\nRun $TO_RUN ..."
ghdl -r $TO_RUN --wave=$TO_RUN.ghw
```

```
return 0
}

if build_and_run; then
 # wyświetlenie przebiegów
 gtkwave $TO_RUN.ghw &
fi

while true; do
 echo -e "\\033[34mAby zakonczyc wpisz e i nacisnij ENTER\\033[0m"
echo -e "\\033[34mAby zakonczyc zachowując pliki tymczasowe wpisz b i nacisnij ENTER\\033[
 0m"
echo -e "\\033[34mAby przbudowac ponownie wcisnij ENTER\\033[0m"

read ctrl;
 if [ "$ctrl" = "e" ]; then
  break
 elif [ "$ctrl" = "b" ]; then
  echo "We don't remove temporary files in $DIR"
  exit
 fi

if build_and_run; then
  if ! jobs 2>&1 | grep gtkwave > /dev/null; then
   gtkwave $TO_RUN.ghw &
  fi
 fi
done

# możnaby usunąć wygenerowane pliki:
# ghdl --remove
# a także robić Makefile (ghdl --gen-makefile) itd

# kończymy sprzątanie
cd /tmp && rm -fr $DIR
```

TESTER.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity tester is
end;

architecture logic of tester is
 signal CLK,RST,a,b,c: std_logic := '1';
 -- uwaga takie przypisanie wartosi poczatkowej nie podlega syntezie (!)
 --          podobnie nie da sie syntezowac opoznien czasowych,
 --          ale to jest testbanch wiec nam to nie przeszkadza
 signal tryb: std_logic_vector (1 downto 0);
 signal wej1, wej2: std_logic_vector (2 downto 0);
 -- w miejscu tym warto wspomnieć o możliwości definiowania aliasów np:
 --    alias pocz_wej1 : std_logic_vector (1 downto 0) is wej1 (1 downto 0);
 -- spowoduje że nazwa pocz_wej1 będzie się odnosiła do dwuelementowego wektora
 -- tożsamego z pierwszymi dwoma elementami wej1
```

```vhdl
begin
  -- zegar
  process
   begin
   wait for 2 ns;
   CLK <= not CLK;
  end process;

  -- zatrzymanie symulacji
  process
   begin
   wait for 100 ns;
   assert FALSE
    report "koniec symulacji"
   severity FAILURE;
  end process;

  RST <= '0', '1' after 0.5 ns;

  a <= '0' after  5 ns, '1' after  25 ns;
  b <= '0' after  9 ns, '1' after  21 ns;
  c <= '0' after 13 ns, '1' after  17 ns;

  tryb <= "00", "10" after 11 ns, "01" after 22 ns;

  mux1: entity work.mux_3to1(z_N_bitowego) port map(MODE=>tryb, a=>a, b=>b, c=>c);
  mux2: entity work.mux_3to1(wybierany_case) port map(MODE=>tryb, a=>a, b=>b, c=>c);
  mux3: entity work.mux_3to1(proces_case) port map(MODE=>tryb, a=>a, b=>b, c=>c);
  mux4: entity work.mux_3to1(wybierany_if_else) port map(MODE=>tryb, a=>a, b=>b, c=>c);
  mux5: entity work.mux_3to1(proces_if_else) port map(MODE=>tryb, a=>a, b=>b, c=>c);

  wej1 <= a&b&c;
  wej2 <= O"5";
  -- zapis 3 bitowej wartosci w notacji oktalnej ...
  -- 8 bitowy wektor w notacji szesnastkowej mozemy wypelnic poprzez X"fa"
  --
  -- z koleji 2#0# oznacza ze pomiedzy # jest zapisana licza stalopozycyjna (integer)
  -- w notacji o podstawie 2

  nor1: entity work.nor3(logic) port map(wej=>wej1);
  nor2: entity work.nor3(proces_while) port map(wej=>wej1);
  nor3: entity work.nor3(proces_loop) port map(wej=>wej1);


  przes1: entity work.rejestr_przesuwny(logic)
   generic map(N=>4)
   port map(D=>a,CLK=>CLK,RST=>RST,ENABLE=>'1');
  -- nalezy wspomniec ze zarowno w generic jak i port map mozemy  uzywac pozycyjnego jak i
  -- nazewniczego sposobu podawania przypisan jednak jezeli jest ich wiecej niz jedno to
  -- sposob poprzez nazwa=>... ("nazewniczy") jest czytelniejszy i wygodniejszy ...
  przes2: entity work.rejestr_przesuwny(modularna)
   generic map(4)
   port map(D=>a,CLK=>CLK,RST=>RST,ENABLE=>'1');
 -- na koniec warto wspomniec o tym iz do tworzenia rozbudowanych testbench'ow mozna
 -- wykorzystac mechanizmy vhdl do obslugi pliku, dzieki czemu pobudzenia testbencha moga
 -- byc umieszczane w zewnetrznym pliku (np. generowanym przez jakies inne oprogramowani)
 -- podobnie ewentualne wyniki zostana zapisane do plikow co umozliwi ich analize
 -- zewnetrznymi narzedziami
end;
```

REJESTR_PRZESUWNY.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity rejestr_przesowny is
 generic (
  N: natural := 1
 );
 port (
  CLK,RST,ENABLE: in std_logic;
  D: in std_logic;
  P_OUT: buffer std_logic_vector (N-1 downto 0)
 );
end;

architecture logic of rejestr_przesuwny is
 begin
  process (CLK, RST)
  begin
   if (RST='0') then
    for i in 0 to N-1 loop
     P_OUT(i) <= '0';
    end loop;
    – sprytniejszym zapisem tego jest:
    –   P_OUT <= (others => '0');
    – korzystamy tu z tzw agregacji umożliwiającej wygodne nadawanie wartości wektorom np.
    :
    –   wektor <= (1|3 => '1', 2|4 =>'0');
    –   wektor <= (1 => '1', 3 => '1', 2|4 =>'0');
    –   wektor <= ('1', '0', '1', '0');
    –   wektor <= "1010"
    – są całkowicie równoważnymi zapisami, jednak pierwszy z nich jest zdecydowanie
    – wygodniejszy dla długich wektorów
    – ponadto dzieki agregacji możemy łatwo wpisać wektor do zestawu sygnałów poprzez:
    –   (skalar, vektor_2, skalar) <= wektor_4
   elsif (CLK='0' and CLK'event) then
    if (ENABLE='1') then
     P_OUT <= D & P_OUT(N-1 downto 1);
     – tak jak do pojedynczych bitow mozemy odwolac sie do zakresow
     – o kolejnosci bitow z zakresu decyduje uzycie to / downto
    end if;
   end if;
  end process;
end;

architecture modularna of rejestr_przesuwny is
 begin
  – uzywamy generate for aby wykorzystac odpowiednia liczbe 1 bitowych multiplekserow
  G0: for i in 0 to N-1 generate
   – mozemy takze warunkowac fragmenty tego co umieszczamy w generate
   G01: if i=N-1 generate
    mux: entity work.przerzutnik(logic)
    – powyzszy zapis oszczedza podawania deklaracji komponentu
    – oraz ewntualnego podawania konfiguracji ...
     port map (D=>D,CLK=>CLK,RST=>RST,ENABLE=>ENABLE,Q=>P_OUT(i));
   end generate;
   G02: if i<N-1 generate
    mux: entity work.przerzutnik(logic)
     port map (D=>P_OUT(i+1),CLK=>CLK,RST=>RST,ENABLE=>ENABLE,Q=>P_OUT(i));
```

```
    end generate;
  end generate;
end;
```

PRZERZUTNIK.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity przerzutnik is
 port (
  D,CLK,RST,ENABLE: in std_logic;
  Q: buffer std_logic
 );
end;

architecture logic of przerzutnik is
 begin
  process (CLK, RST)
  begin
   if (RST='0') then
    Q<='0';
   -- reagowanie na opadajace (CLK='0') zbocze zegara (CLK'event)
   -- powoduje utworzenie przezutnika a nie zatrzasku
   -- ktory uzyskalibysmy piszac: if (ENABLE='1') then Q<=D; end if;
   elsif (CLK='0' and CLK'event) then
    if (ENABLE='1') then
        Q<=D;
    end if;
   end if;
  end process;
end;
```

NOR.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity nor3 is
 port (
  wej: in std_logic_vector(0 to 2);
  wyj: out std_logic
 );
end;


architecture logic of nor3 is
 begin
  wyj <= not (wej(0) or wej(1) or wej(2));
  -- ze wzgledu na to iz jej funkcjonalnosc opisuje sie w taki prosty sposob
  -- w praktycznych zastosowanuiach raczej nie ma sensu robienia z niej osobnego komponentu ..
  .
end;
```

```
architecture proces_while of nor3 is
 begin
  process
   variable i: integer range -1 to 3;
   variable x: std_logic;
   begin
    – czekamy na zmiane ktoregos z sygnalow
    wait on wej;
    – jest to podobne do listy podawanej po slowie proces ...
    – tyle ze wtedy oczekiwanie jest umieszczane na kocu ciala procesu ...
    – nalezy wspomniec ze jest tez wait until (czekanie na spelnienie warunku)

    x := '0';
    i := -1;

    – petla typu while
    L0: while i<2 loop
     i := i+1;

     – instrukcja next powoduje przejscie do nastepnego kroku petli
     next when wej(i) = '0';
     – pomiedzy next a when mozna podac etykiete petli ktorej iteracje chemy zakonczyc ...

     x := '1';
    end loop;

    wyj <= not x;
  end process;
end;


architecture proces_loop of nor3 is
 begin
  process (wej)
   variable i: integer range -1 to 2;
   variable x: std_logic;
   begin
    x := '0';
    i := -1;
    – petla prosta
    L0: loop
     exit when i=2;
     i := i+1;
     next when wej(i) = '0';
     x := '1';
     exit;
    end loop;

    wyj <= not x;
  end process;
end;
```

MULTIPLEXER.VHDL

```
library ieee;
use IEEE.STD_LOGIC_1164.all;
```

```
-- deklarujemy multiplexer 1 bitowy o M wejsciach
entity multiplexer is
  -- parametr ogolny, modyfikuje zachowanie jednostki projektowej
  -- na etapie jej umieszczania w projekcie (cos jak #define w C)
  generic (
    M: natural := 2
  );
  -- porty wejsciowe i wyjsciowe jednostki
  -- przy ich pomocy komunikujemy sie z innymi elementami systemu
  -- kierunek portu oprocz zaprezentowanego in i out moze byc:
  --   * inout - port dwukierunkowy (jak GPIO w AVR)
  --   * buffer - port wyjsciowy z ktorego mozemy odczytac wpisana do niego
  --              wartosc (przerzutnik wyjsciowy),
  --              tryb ten może stwarzać problemy z kompatybinością
  port (
    -- tryb jest liczba calkowita z zakresu 0-M
    MODE: in integer range 0 to M-1;
    -- wejscia zgrupowane sa w postaci wektoru M-1 bitowego
    DATA_IN: in std_logic_vector (M-1 downto 0);
    -- wyjsciem jest pojedynczy bit
    DATA_OUT: out std_logic
  );
end;

-- definiujemy architekture naszego multipleksera
architecture logic of multiplexer is
  begin
    DATA_OUT <= DATA_IN(MODE);
end;
```

MULTIPLEXER_3TO1.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

-- uzywamy mojego_pakietu
use work.moj_pakiet.all;


-- deklarujemy multiplexer 3 wejsciowy
entity mux_3to1 is
  port (
    -- tryb jest wartoscia 2 bitowa
    MODE: in std_logic_vector (1 downto 0);
    -- mamy 3 wejscia
    a,b,c: in std_logic;
    -- i jedno wyjscie
    DATA_OUT: out std_logic
  );
end;


-- definiujemy architekture naszego multipleksera
architecture z_N_bitowego of mux_3to1 is
  -- gdy chcemy wykorzystac jakis element musimy przytoczyc
  -- (lekko zmodyfikowana) tresc jego entity w postaci opisu komponentu:
```

```
 component mux is
  generic (
   M: natural := 4
   – nadpisujemy domyslna ilosc bitow
  );
  port (
   MODE: in integer range 0 to M-1;
   DATA_IN: in std_logic_vector (M-1 downto 0);
   DATA_OUT: out std_logic
  );
 end component;

– deklaujemy wewnetrzne pomocnicze sygnaly
 signal tryb: integer range 0 to 2;
 signal wejscie: std_logic_vector (3 downto 0);

begin
  wejscie <= '-' & c & b & a;
  – musimy podac tez najwyzszy bit (aby jego bitowosc byla 2^N)
  – jako ze nie jest on nam potrzebny a korzystamy z std_logic to podajemy ”don’t cary”

  – przy pomocy funkcji to_int z moj_pakiet konwertujemy wartosc binarna na integer
  tryb<=to_int(MODE);

  – mapujemy odpowiednie wejscia i wyjscia dla egzemplarza komponentu multiplexer
  – okreslonego etykieta mux_31
  mux_31: mux port map (MODE=>tryb, DATA_IN=>wejscie, DATA_OUT=>DATA_OUT);
  – etykiety mozemy podac w kazdej linii vhdl, ale tylko w niektorych wypadkach jest
  – to wymagane (wlasnie taka sytuacja ma tutaj miejsce)
end;


– mozna takze okreslic konfiguracje komponentow uzywanych do budowy architektury
configuration mux_3to1_config of mux_3to1 is
 for z_N_bitowego – dla architektury ”z_N_bitowego”
  for mux_31: mux use entity
   work.multiplexer(logic);
  end for;
  – tu muglbybyc kolejny for dla kolejnych komponentow
 end for;
 – tu muglby byc kolejny for dla kolejnej architektury
end;
– gdyby w architekturze zamiast mux podac nazwe komonentu zgodna z jego entity
– to blok ten bylby niepotrzebny bo uzyta bylaby konfiguracja domyslna


– architektur moze byc kilka ...
architecture wybierany_case of mux_3to1 is
 begin
  with MODE select DATA_OUT <=
   a when "00",
   b when "01",
   c when "10",
   '-' when others;
   – w przypadku std_logic formalnie (część kompilatorów tego nie wymaga)
   – praktycznie zawsze konieczne jest podawanie ”when others”
   – gdyż mamy inne niż 0 i 1 wartości sygnału
end;

architecture proces_case of mux_3to1 is
```

```
 begin
  process (a,b,c,MODE)
   begin
    case MODE is
     when "00" => DATA_OUT <= a;
     when "01" => DATA_OUT <= b;
     when "10" => DATA_OUT <= c;
     when others => DATA_OUT <= '-';
    end case;
  end process;
end;

-- UWAGA ze wzgledu na kolejnosc wykonywania instrukcji w ponizszych konstrukcjach
-- oba uklady opisane ponizej moga sie zsyntezowac jako hierarchiczna struktura
-- multiplekserow 2 wejsciowych
--
-- NIE JEST to zatem zalecany opis multipleksera 3 na 1
architecture wybierany_if_else of mux_3to1 is
 begin
  DATA_OUT <=
   a when MODE = "00" else
   b when MODE = "01" else
   c when MODE = "10" else
   '-';
end;

architecture proces_if_else of mux_3to1 is
 begin
  process (a,b,c,MODE)
   begin
    if MODE="00" then
     DATA_OUT <= a;
    elsif MODE="01" then
     DATA_OUT <= b;
    elsif MODE="10" then
     DATA_OUT <= c;
    else
     DATA_OUT <= '-';
    end if;
  end process;
end;
```

MOJ_PAKIET.VHDL

```
-- uwidaczniamy biblioteke IEEE
library ieee;

-- uzywamy z tej biblioteki wszystkich elementow z pakietu std_logic_1164
-- pozwala to na pisanie std_logic zamast ieee.std_logic_1164.std_logic
use ieee.std_logic_1164.all;
-- oraz z pakietu numeric_std
use ieee.numeric_std.all;


-- deklarujemy nasz wlasny pakiet
package moj_pakiet is
 -- w ramach niego bedzie funkcja to_bits()
```

```
  function to_bits(liczba, dlugosc_wektora : in integer) return  std_logic_vector;
  -- oraz to_int()
  function to_int(wartosc_bitowa : in std_logic_vector) return integer;
  -- procedury w odruznieniu od funkcji moga modyfikowac przekazane do nich argumenty ...
  procedure zwieksz(dane: inout std_logic_vector; ovf: out std_logic);
end package;

-- definiujemy zawartosc naszego pakietu
package body moj_pakiet is
  function to_bits(liczba, dlugosc_wektora : in integer) return  std_logic_vector is
    begin
      return std_logic_vector(to_unsigned(liczba,dlugosc_wektora));
  end function;

function to_int(wartosc_bitowa : in std_logic_vector) return integer is
    begin
      return to_integer(unsigned(wartosc_bitowa));
  end function;

procedure zwieksz(dane: inout std_logic_vector; ovf: out std_logic) is
    begin
      -- petkla for po calym zakresie binarnym wektora wejsciowo/wyjsciowego
      -- to cos po ' nazywa sie atrybutem ...
      -- xx'range zwraca zakres xx'left to xx'right
      -- xx'right jest maksymalnym indeksem wektora
      -- ...
      F1: for i in dane'range loop
        -- przypisanie do zmiennej
        dane(i) := not dane(i);
        -- przerwanie petli (tak jak break w C)
        exit when dane(i) = '1';
        -- pomiedzy exit a when mozna podac etykiete petli z ktorej chemy wyjsc ...

        -- jezeli petla doszla do konca
        -- nieosiagajac warunku przerwania to mamy przepelnienie
        if i = dane'right then
          ovf := '1';
        else
          -- galaz else jest obowiazkowa gdy nie chemy tworzyc zatrzaskow
          ovf := '0';
        end if;
      end loop;
  end procedure;
end;
```

AUTOMAT.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity automat is
  port (
    wejscia: in std_logic_vector (1 downto 0);
    clk, reset: in std_logic;
    wyjscia: out std_logic_vector (1 downto 0)
  );
end;
```

```
architecture logic of automat is
 type STANY is (
  S0,
  S1,
  S2
);

signal STAN_OBECNY, STAN_NASTEPNY: STANY := S0;

begin
 process (clk, reset)
 begin
  if (reset='1') then
   STAN_OBECNY <= S0;
  elsif (clk='1' and clk'event) then
   STAN_OBECNY <= STAN_NASTEPNY;
  end if;
 end process;

process (STAN_OBECNY, wejscia)
 begin
  case STAN_OBECNY is
   when S0 =>
    wyjscia <= "01";
    -- wyjscia zaleza tylko od stanu wiec automat Moora

    if (wejscia = "01") then
     STAN_NASTEPNY <= S1;
    else
     STAN_NASTEPNY <= S2;
    end if;
   when S1 =>
    wyjscia <= "11";

    if (wejscia = "01" or wejscia = "10") then
     STAN_NASTEPNY <= S2;
    elsif (wejscia = "11") then
     STAN_NASTEPNY <= S1;
    else
     STAN_NASTEPNY <= S0;
    end if;
   when S2 =>
    wyjscia <= "00";

    if (wejscia(0) = '0') then
     STAN_NASTEPNY <= S1;
    else
     STAN_NASTEPNY <= S2;
    end if;
  end case;
 end process;
end;
```

Więcej na temat programowania

Narzędzi programistyczne i skryptowe

Ważnym elementem są kompilatory (gcc), oraz interpretatory języków skryptowych (gawk, perl, php, python). Umożliwiają one tworzenie programów ze źródeł jak i uruchamianie programów tworzony w językach skryptowych. Język awk jest szczególnie przydatnym narzędziem przetwarzania plików tekstowych i jako taki mógłbybyć zaliczony do opisywanych użytków tekstowych.

- gcc[1098] - kolekcja kompilatorów GNU (m.in. C i C++)
- clang[1099] - alternatywny dla gcc kompilator C i C++
- gawk[1100] - wersja AWK z projektu GNU
- perl[1101] - interpreter języka skryptowego
- php[1102] - interpreter języka skryptowego
- python[1103] - interpreter języka skryptowego
- strace[1104] - narzędzie do śledzenia wywołań funkcji dokonywanych przez monitorowany program
- - utworzenie statycznego pliku wykonywalnego w oparciu o detekcję wykorzystywanych bibliotek dynamicznych
- subversion[1105] - systemy zarządzania wersjami
- git[1106] - systemy zarządzania wersjami
- bzr[1107] - systemy zarządzania wersjami
- hping[1108] - programik pozwalający na operowanie gołymi pakietami i nie tylko ...
- netcat6[1109] (gnu netcat, netcat) - programiki pozwalające na operowanie wysyłanie pakietów TCP i UDP z definiowaną przez nas zawartością, umożliwia m.in. testowanie usług sieciowych (takich jak smtp, www, jabber, ...), więcej w tutaj
  Homepage: `http://netcat.sourceforge.net/http://www.vulnwatch.org/netcat/`
- `socat` - SOcket CAT - uniwersalne narzędzie do dwukierunkowej wymiany danych (umożliwia m.in. przekierowanie portu szeregowego poprzez sieć[1110])
- tcputils[1111] - zestaw narządzi ułatwiających korzystanie z TCP w skryptach ...
- firebird[1112] - biblioteka bazy danych (Poznaj Firebird w 2 minuty[1113])
- - kompilator C++
- - narzędzia ułatwiające tworzenie paczek Debianowych
- - kompilator i biblioteka standardowa C dla AVR
- - interp[reter PHP działający z linii poleceń
- - program do projektowania GUI w Qt
- - IDE dla KDE
- - interpreter języka skryptowego
- - konwersja pomiędzy XML, HTML, CSV i formatem liniowym
- - XSLT processor
- - XSLT processor
- - XSLT processor
- - XML-to-any converter
- treeline[1114] - graficzna obsługa drzewiastych baz danych, w szczególności plików XML

Perl

Perl[1115] jest wszechstronnym językiem programowania, umożliwiającym programowanie w różnych modelach. Posiada bardzo dużą liczbę dodatkowych modułów (bibliotek) zgromadzoną w archiwach CPAN[1116]. Jest jednym z najpopularniejszych języków używanych do tworzenia CGI. Z Perla wywodzą się takie języki jak: PHP[1117], Ruby[1118] i Python[1119].

Do ewidentnych zalet tego języka należy wspomniana już kolekcja gotowych modułów (których wykorzystanie zdecydowanie upraszcza tworzenie programu), dynamiczne typy zmiennych, wbudowane wyrażenia regularne, tablice asocjacyjne. Niektórzy jako zaletę perla podają także "naturalny sposób zapisu" uzyskiwany przez wykorzystywanie zmiennej wbudowanej `$_`, która jest argumentem domyślnym wielu funkcji. Rozwiązanie to ma jednak wielu przeciwników, zwłaszcza osoby które tak jak ja lepiej czują się w językach formalnych niż naturalnych.

Perl uważany jest za język skryptowy, jednak do końca nim nie jest. Kod skryptu przed jego wykonaniem

poddawany jest kompilacji do kodu pośredniego. Istnieje możliwość wykonania takiej kompilacji i wykorzystywania (oraz dystrybucji) kodu pośredniego.

PODSTAWY

Komentarzem jest wszystko od znaku # do końca linii (uwaga: znak # może występować także w ramach jakiś operatorów - nie oznacza on wtedy komentarza).

Łańcuchy znaków umieszczane są w ramach `""` (wnętrze napisu podlega interpretacji) lub `''` (wnętrze napisu nie podlega interpretacji). Napisy można także ograniczać przy pomocy otwierających `qqX` lub `qqX` (gdzie `X` jest znakiem ogranicznika) i zamykającego znaku ogranicznika, jeżeli znak ogranicznika jest którymś z nawiasów otwierających na końcu napisu używa się odpowiedniego nawiasu zamykającego. Do łączenia napisów służy operator `.` (kropka).

ZMIENNE

Nazwa zmiennej w perlu poprzedzana jest znakiem oznaczającym jej typ są to:

- `$` - zmienne skalarne (liczby, napisy, ...)
- `@` - tablice zwykłe
- `%` - tablice asocjacyjne
- `&` - podprogramy
- `*` - zmienne globalne

Standardowo nie ma potrzeby deklarowania zmiennych - perl gdy napotka nieznaną zmienną sam utworzy "pustą" zmienną (o wartości zero/napis pusty/...). Można to zmienić umieszczając w kodzie `use strict 'vars';`, wymusza to jawne deklarowanie zmiennych ze słowem kluczowym `my`, dodanie tego słowa do deklaracji zmiennej powoduje iż zmienna staje się zmienną lokalną (domyślnie wszystkie zmienne są globalne.

ZMIENNE PRZEDEFINIOWANE

- `$_` - argument domyślny
- `@_` - tablica argumentów domyślnych
- `@ARGV` - tablica argumentów przekazanych do skryptu
- `%ENV` - tablica zmiennych środowiskowych
- `$?` - status wyjścia z funkcji system()
- `$$` - PID wykonywanego skryptu

Zobacz w Sieci: Perl @ Wikibooks[1120], Perl 6 Programming[1121]

Zachęcam także do zapoznania się z kilkoma przykładami programów stworzonych przeze mnie w perlu - smtp2xmpp.pl (nowsza wersja klienta jabbera przeznaczona do wysyłania komunikatów o nowej poczcie), mail-bot.pl (bot jabberowski odpowiadający na pytania o nową pocztę - w programie pokazane deklarowanie funkcji, korzystanie z daty i czasu oraz zewnętrznych aplikacji).

LUA

Lua[1122] jest interesującym językiem skryptowym, którego dodatkowym atutem jest dobra współpraca z C (skrypty Lua mogą służyć do zapewnienia w wygodny sposób warstwy skryptowej programom tworzonym w C/C++). Przykład współpracy Lua i C pokazany jest w c_lua_api.lua i c_lua_api.c (warto zapoznać się także z użytą w tym przykładzie biblioteczką LuaGenericCall[1123]). Poniżej prezentuje przykład bezpośredniego wykorzystania Lua jako języka skryptowego (uruchamia się go `lua5.1 nazwa_pliku.lua`). Więcej na temat tego języka w Lua 5.1 Reference Manual[1124], Lua @ Wikibooks[1125]Lua-users wiki - Sample Code[1126].

```
– w ten sposob tez mozna deklarowac funkcje ...
printf = function(fmt, ...)
 – ten trzykropek powoduje ze jest wieloargumentowa
 – uwaga: to dziala w lua >= 5.1
 return io.write(fmt:format(...))
end
```

```
printf("Ala w dniu %s miała %d koty\n", os.date("!%F %T (UTC)"), 3)
-- wykrzyknik w napisie formatujacym date powoduje zwrocenie daty w UTC
-- napisy formatujace sa zgodne z strftime()

napisA="Kot ma Ala abcabcbbba"
napisB=string.format("HELLO x %d", 3.5)


napisC, num = string.gsub(napisA, "b.a", "")
-- druga zwrocona wartoscia jest ilosc podmian - odbieramy oba agrumenty
print( napisC )
print( "zastapien:", num )
print( string.gsub(napisA, "(K[^ ]*)(.*)(A[^ ]*)", "%3 %2 %1") )

-- mozemy zobie dynamicznie przenazywac funkcje
my_print = printf
my_print("AAA %s BBB\n", napisB)

-- uwaga: tablice indeksowane numerycznie indeksowane sa od 1
-- #tablica zwraca ilosc elementow (numer ostatniego)
tablica = { "Q", "W", "E", "R" }
print(#tablica, tablica[1])

-- dodajemy element na koncu i to innego typu (!)
tablica[#tablica+1]=1
print(#tablica, tablica[5])

-- sa tez struktury ...
struktura = { a="Q", b="W" }
print(struktura["a"], struktura.b)

struktura.c = tablica
print(struktura.c[3])

-- w zasadzie to tak jak tablica tez rodzaj tabeli tylko z bardziej dowolnie nazywanymi polami
struktura["ala"] = "kota\n"

-- mozemy tez dodac metode jako skladowa
struktura.wypisz = printf
struktura.wypisz("Ala ma %s", struktura["ala"])

-- więcej na http://www.lua.org/manual/5.1/
```

C_LUA_API.LUA

```
-- funkcja ktora bedziemy wywolywac z C
function jakas_funkcja(val)
 print("To jakas funkcja", val, tabelka[1])
 return 2*val
end

-- wypisanie tabelki przekazanej z C
print "Wyglad tabelki:"
for index, value in pairs(tabelka) do
  print(index, value, type(value))
end
```

```
– wywolanie funkcji zdefiniowanej w programie w C
print("Funcka z C zwrocila:", funkcja_z_c(12, 4))

– zwrocenie wynikow do C
return 23, 20.34, "Ala ma kota"
```

Ruby

Ruby[1127] jest bardzo ciekawym interpretowanym językiem skryptowym. Pozwala m.in. na: korzystanie z tablic (zarówno zwykłych jak i asocjacyjnych), łatwą konwersję między tablicami a napisami i na odwrót, wygodne operacje napisowe, korzystanie z funkcji o nie ustalonej z góry liczbie argumentów, korzystanie z wyrażeń regularnych, programowanie obiektowe. Poniżej przedstawiam przykład prezentujący podstawowe cechy tego języka, zachęcam także do zapoznania się z Ruby @ Wikibooks[1128].

```
#!/usr/bin/ruby

ilosc_kotow=1
ilosc_kotow=ilosc_kotow+1
print "Ala ma #{ilosc_kotow} kotow:\n"
print "KOTY: " + " mial" * ilosc_kotow + "\n"
print "A gdy znajdzie nasteopnego bedzie miala #{ilosc_kotow+1} kotow\n"

# jak powyżej widać napisy możemy dodawać a nawt mnożyć przez liczbę
# w ramach napisu możemy wypisywać zmienną a także wynik operacji matematycznej na zmiennej

# deklaracja funkcji, pierwszy argument umieszczony będzie w zmiennej napis,
# drugi i wszystkie kolejne w elementach tablicy ile
# można także przypisać domyślne wartości poprzez podanie =wartosc
def wypisywanie(napis, *ile)
 # pętla for po liczbach z zakresu
 for i in (2..ile[0])
  print i
end
 print "\n"

# sprawdzamy czy tablica ile ma więcej niż 1 element
 if ile.size > 1
  # pętla while
  while (ile[1] > 0)
   ile[1]=ile[1]-2
   print napis + "\n"
  end
 end
end

# wykorzystanie funkcji - 3 argumenty
wypisywanie("Witaj świecie", 5, 6)

# wypisanie odzielacza - warto zauważyć że pierwszy \n będzie przytoczony dosłownie
# gdyż użyte zostały pojedyńcze sudzysłowia
print '==========\n==========' + "\n"

# kolejne wykorzystanie funkcji - 2 argumenty
wypisywanie("Helo World", 5)

# tablica - deklarowanie, łączenie, zamiana na napis
```

```
tablica = [2, "4", 6]
print "Elemnt -2 to: " + tablica[-2] + "\n"
# uwaga: takie łączenie działa dlatego że ten element jest napisem ;-)
tablica = tablica + ["abc", "def"]
tablica = tablica * 2
print tablica.join(" - ") + "\n"

# zakres tablicy (można go używać np. w pętli for po in)
print tablica[0..2]
print "\n"

# przy pomocy zmienna_napisowa.split("podzial") możemy podzielić zmienną napisową
# na elementy tablicowe względem zadanego napisu
napis="Ala , ma , kota, albo , dwa"
tablica=napis.split(" , ")
print tablica.join("/") + "\n"

# tablice asocjacyjne
tablica_asocjacyjna = {1 => 2, "drugi" => 4, 2 => "kot"}
print "-==-#{tablica_asocjacyjna["drugi"]}-==-#{tablica_asocjacyjna[5]}-==-\n"
```

Tcl/Tk

Tcl[1129]/Tk[1130] jest skryptowym językiem umożliwiającym budowanie graficznych interfejsów użytkownika (biblioteka Tk). Samo Tcl jest językiem komend tak jak typowe unixowate języki skryptowe takie jak sh. Tk jest z kolei biblioteką odpowiedzialną za interfejs graficzny, jest ona dostępna także dla wielu innych języków programowania, jednak pierwotnie związana jest właśnie z Tcl. Poniżej przedstawiam przykładowy skrypt obrazujący użycie Tlc/Tk. Zobacz w sieci: Tcl/Tk Tutorial[1131].

```
#!/usr/bin/wish

label .info_label -text "To testowy programik"
pack .info_label

# deklaracja pola wczytywania/wypisywania tekstu
entry .pole_tekstowe
pack .pole_tekstowe

# funkcja z jednym argumentem
proc dodawanie { ctrl } {
 # deklaracja pustego napisu
 set napis ""

# warunek if
 if { $ctrl == 2 } {
  # wypisanie (na terminal) wyniku dodawania
  puts [expr $ctrl+5]
 } elseif {$ctrl == 1} {
  set lista [list "Ala " {ma } kota]
  # pętla forech - for jak w bashu ;-)
  foreach wyraz $lista {
   set napis "$napis$wyraz"
  }
  # podnapis na terminal
  puts [string range $napis 3 end]
 } else {
  # pętla for
  for { } {$ctrl>0} { set ctrl [expr $ctrl-1] } {
```

```
    # wypisanie napisu
    puts "Hi hi hi"
  }
 }
 # wypisanie $napis w gui
 .pole_tekstowe insert end $napis
}

# funkcja z dwoma argumentami
proc odejmowanie { mode ile } {
 switch $mode {
  "pierwszy" {
   while { $ile >= 10 } {
    puts $ile
    set ile [expr $ile - $ile/2]
   }
  }
  "drugi" {
   # odczyt pliku
   set input_file [open "/etc/passwd" r]
   set zawartosc_pliku [read $input_file]
   close $input_file
   # wypisanie zawartości pliku
   puts $zawartosc_pliku
   # rozłożenie zawartości na linijki i wypisanie jednej z nich
   set linijki [split $zawartosc_pliku "\n"]
   puts [lindex $linijki 2]
   # wykorzystane tu listy mają jeszcze parę możliwości ...
  }
  default {
   set "I co ja mam robić ?"
  }
 }
}

# tworzymy przycisk
button .button1 -text "Dodaj 1" -command "dodawanie 1"
button .button2 -text "Dodaj 2" -command "dodawanie 2"
button .button3 -text "Dodaj 3" -command "dodawanie 3"
button .button4 -text "Odejmij 1" -command "odejmowanie pierwszy 100"
button .button5 -text "Odejmij 2" -command "odejmowanie drugi 10"
button .button6 -text "Odejmij 3" -command "odejmowanie trzeci"

# wyświetlamy przyciski
pack .button1
pack .button2
pack .button3
pack .button4
pack .button5
pack .button6
```

Erlang

Erlang[1132] jest dość oryginalnym językiem programowania. Jest to język wybitnie funkcyjny[1133] (do tego stopnia iż nie ma w nim pętli), posiada dość nietypową składnie, ale np. nie wymaga deklarowania zmiennych.

Wiele z funkcji języka możemy testować w jego linii poleceń (`erl`, jednak wiele z nich (np. deklarowanie funkcji) wymaga aby były umieszczone w modułach - w tym celu tworzymy plik nazwamodulu.erl, umieszczamy w

nim nasz kod a w linii poleceń wydajemy `c(nazwamodulu).` co powoduje przebudowanie modułu, po czym możemy korzystać z zdefiniowanych w nim funkcji - `nazwamodulu:funkcja(argumenty)`.

Poniżej prezentuję mój patch (aktualnie dodany już do głównego kodu) umożliwiający przechowywanie wiadomości offline dla użytkowników autoryzowanych przez PAM w serwerze XMPP "ejabberd". Polega on na zastąpieniu trywialnego ciała funkcji "is_user_exists" w module "ejabberd_auth_pam" następującym kodem:

```
is_user_exists( _User, _Server ) ->
 %% wczytujemy do FileStr plik /etc/passwd dodając na początku pustą linię
 {ok,FileBin}=file:read_file("/etc/passwd").
 Tmp=binary_to_list(FileBin).
 FileStr=string:concat("\n", Tmp).
 %% tworzymy RegExp dodając do zmiennej napisowej _User
 %% znak nowej linii na początku i : na końcu
 Tmp2=string:concat("\n", _User).
 RegExp=string:concat(Tmp2, ":").
 %% jeżeli w FileStr wystąpiło wyrażenie RegExp
 %% zwracamy true w przeciwnym razie false
 case regexp:match(FileStr, RegExp) of
   {match, _, _}  ->
       true;
   _ ->
       false
 end.
```

Nie jest to rozwiązanie eleganckie (odwołania do /etc/passwd to nie PAM), ale w większości wystarczające. Kod ten pokazuje korzystanie z plików i napisów w erl'angu. Obrazuje również zdefiniowanie funkcji przyjmującej dwa argumenty i zwracającej wartość true lub false, deklaracja ciała funkcji następuje po `->`, a kończy się kropką.

W ramach serwisu opublikowane jest kilka patchy mojego autorstwa do serwera "ejabberd", który stworzony jest właśnie w erlangu - zachęcam do zapoznania się z nimi jako przykładami programowania w tym języku - "ejabberd_log_offline_message.diff" (łatka stanowiąca fragment system powiadomień XMPP/SMTP, pokazuje obsługę plików, operacje na napisach, a także tworzenie katalogu i formatowanie czasu), JID aliases (zestaw łatek implementujących aliasy JID, pokazuje komunikację z zewnętrzną aplikacją, instrukcję warunkowe, tworzenie funkcji i zwracanie z nich wyników.

Zobacz w Sieci: dokumentacja języka[1134].

GNU Octave

GNU Octave[1135] jest środowiskiem obliczeń kompatybilnym z MatLabem. Poniżej kilka podstaw dotyczących analizy danych w tym środowisku (tworzenie histogramu):

```
# ładujemy plik z danymi
load wynik.dat
# rysowanie zwykłego wykresu (x z kolumny 1, y z kolumny 3
plot(wynik(:,1),wynik(:,3));
title("tytul wykresu");
# histogram w oparciu o próbki 1-25000 z kolumny 9
# przedział histogramowania -400000 do 400000
# pojedynczy słupek histogramu o szerokości 1000
hist(wynik(1:25000,9),-400000:1000:400000);
# opis osi
xlabel("czas trwania [ns]");
ylabel("ilosc przypadkow");
# zapis do pliku ps
print("histogram1.ps", "-dps");
```

Podstawy operowania na zmiennych:

```
% wszystko od znaku % do końca linii to komentarz
% przypisanie wartości do zmiennej
```

```
a=2*3+1
% liczba zespolona (bez echo na ekran - zapewnia to średnik po komendzie)
% uwaga: gdy 10*i to nie zadziała jako zespolone gdy zdefiniowaliśmy zmienna i
z=a+10i;
% część rzeczywista i urojona liczby zespolonej
real (z)
imag (z)
% zmienna napisowa
l='Ala ma kota'
% wczytanie zmiennej z klawiatury
x=input('napisz: ');
% macierz losowa o wymiarach  2x3 (2 wiersze 3 kolumny)
m=rand(2,3);
% wyświetlenie
disp(m)
disp(sprintf('To %d', real (z)))
% informacje o zmiennych
who
whos
% kasowanie zmiennych
clear m
clear
% czyszczenie ekranu ...
clc
```

Podstawy operowania na macierzach:

```
% macierz z zerami o wymiarach 5x3
m=zeros(5,3)
% macierz wypełniona jedynkami
m=ones(4);
% macierz losowa o rozkładzie normalnym
m=randn(4)
% jawna deklaracja zawartości macierzy
m=[1,2,3;4 5 6]
% macierz z elementami podniesionymi do kwadratu
m .* m
m.^2
% macierz transponowana
m'
% cala 2 kolumna
m(:,2)
% cały pierwszy wiersz
m(1,:)
% drugi wiersz, kolumny 2 i 3
m(2,2:3)
max(m) % wektor z maksymalnymi wartościami z kolumn
max(max(m)) % == max(m(:))
% podobnie min i sum
% wybór w oparciu o warunek
find(e>3)
% macierz logiczna na podstawie warunku i macierzy
c = (e>3)
% macierz pustawa ...
f = sparse ([4,5], [6,6], [2,3]) % w 6 kolumnie w wierszu 4 wartość 2 a w 5 wartość 3
full(f) % pełne wypisanie macierzy f - z zerami
```

— tworzenie macierzy diagonalnej - diag
— górny/dolny trójkąt z macierzy - `tril` / `triu`

— ślad macierzy - `trace`
— wyznacznik macierzy - `det`
— odwrócenie macierzy - `inv`

Pętle i instrukcje warunkowe:

```
x =0;
% while
while x<5
        x = x+1;
end
% for
for x=-5:0.1:2
        a=x^2
end
% mamy break i return ...
% if
if x<5
        a=1
else if x<8
        a=2
else
        a=3
end
% switch
switch x
        case 1
                a=2
        case 3
                a=1
        otherwise
                a=0
end
```

Obsługa plików binarnych:

```
plik = fopen("'nazwa', 'rb')
autor =fread(plik,80,'char')
char(autor)
cahr(autor(1)) ...
dzien=fread(plki,1,'int')
% aby cofnąć / przejść do przodu (o 20):
fseek(plik,20,0)
pomiar=fread(plik,1000,'int')
```

Inne przydatne informacje:

— pomoc na temat funkcji f - `help f`
— deklarowanie funkcji - funkcje stanowią pliki .m, wszystkie komentarze do pierwszej pustej linii stanowią dokumentację (treść pomocy), można dodatkowo skorzystać z konstrukcji `function [suma,iloczyn] = zad_30licz(n1, n2, plik)` aby zadeklarować wektor wyjściowy oraz argumenty funkcji
— deklarowanie funkcji ("inline") - `suma = @(x,y) x+y;`

— znajdowanie minimów funkcji sin(x) dla zakresu od 0 do 3 - `fminbnd('sin(x)', 0, 3)`
— znajdowanie miejsca zerowego funkcji, szukanie zaczynamy od 5- `fzero('sin(x)', 5)`
— znajdowanie miejsc zerowych wielomianu - `p=[-3,0,1,0,-1]; roots(p); % -3x^4 + x^2 - 1`

— suma elementów wektora w - `sum(w)`
— iloczyn elementów wektora w - `prod(w)`
— średnia elementów wektora w - `mean(w)`

— odchylenie standardowe elementów wektora w - `std(w)`

— interpolacja - `interp1`
— aproksymacja (wielomianowa) - `polyfit()` oraz `polyval()`
— eliminacja Gaussa - `lu()`

— pomiar czasu - wyzerowanie i włączenie stopera - `tic`
— pomiar czasu - wypisanie wartości stopera - `toc`

Zobacz w Sieci: Matlab @ FUW[1136]

TeX i LaTeX - porady

TeX jest komputerowym systemem profesjonalnego składu drukarskiego. Głównym elementem systemu jest oczywiście sam program TeX, który zajmuje się przetwarzaniem instrukcji umieszczonych w tworzonym dokumencie. Podstawowy zestaw poleceń jest dość ubogi, jednak pozwala (co prawda nie zawsze w prosty sposób ...) zrobić niemalże wszystko co byśmy chcieli (w odróżnieniu od innych systemów DTP nie ogranicza on nas ...). Dużym ułatwieniem są tutaj makra, które możemy tworzyć zarówno sami, jak również skorzystać z gotowych zestawów takich LaTeX, czy ConTeXt.

Oprócz klasycznego kompilatora tworzącego pliki dvi[1137] jakim jest `tex` (`latex`) powtały także inne, nowocześniejsze kompilatory (La)TeX'a takie jak `pdflatex` (tworzący bezpośrednio pliki pdf), `xelatex` (natywna obsługa pełnego unicode i czcionek OpenType[1138], generowanie plików xdv), `lualatex` (natywna obsługa pełnego unicode i czcionek OpenType[1139], bezpośrednie generowanie plików pdf)

Poniżej przedstawiam:

— przykłądową klasę dokumentu - office.cls, która oprócz ustawień związanych z dokumentem dostarcza makra ułatwiające tworzenie odwołań do punktów w ramach list numerowanych "easylist" oraz tworzenie tabel z wyrównywanymi w pionie i poziomie komórkami
— office-demo.tex - demonstracyjny dokument bazujący na office.cls demonstrujący niktóre z możliwości systemu LaTeX wraz z wybranymi pakietami makr.

Efekt kompilacji office-demo.tex można zobaczyć tutaj[1140]. Po bardziej szczegółowe omówienia odsyłam do: Nie za krótkie wprowadzenia do systemu LaTeX 2e[1141], dokumentacji poszczególnych pakietów i innych źródeł.

OFFICE.CLS

```
% INSTALACJA
% w ~/.bashrc dodać:
%  export TEXINPUTS=.:~/.tex:
% utworzyć dowiązanie symboliczne w ~/.tex do tego pliku
%
% alternatywnie umieszczamy w własnym katalog w /usr/share/texlive/tex/latex
% a następnie jako root wykonujemy: mktexlsr
%
% UŻYCIE:
%  w związku z lepszym wsparciem utf8 w lualatex i usunięciem z dokumentu
%   \RequirePackage[utf8]{inputenc}
%  do generowania dokumentów opartych na tej klasie rekomenduje wywołanie:
%   lualatex --shell-escape PLIK.tex


%
% INFORMACJE O KLASIE I PODSTAWOWE DEKLARACJE
%

\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{office}[2015/03/16 v1.0 Dokument biurowy]
```

```
%
% ŁADOWANIE PODSTWOWYCH PAKIETÓW
%

% włączamy pakiety umożliwiające obsługę opcji typu klucz=wartosc w
\RequirePackage{kvoptions}   % klasach i pakietach
\RequirePackage{xkeyval}     % w komendach

% włączamy pakiety obsługujące przydatne narzędzia itp
\RequirePackage{xstring}        % manipulacje napisami - m.in. \StrBefore, \StrBetween, ⤸
\StrBehind
\RequirePackage{etoolbox}       % \patchcmd, \ifdefstring, \ifstrequal, \ifdefempty
\RequirePackage{xparse}         % \DeclareDocumentCommand (eg. two optional args with defaults)
\RequirePackage{catchfile}      % load file into register
\RequirePackage{calc}           % \widthof{TEXT}, \maxof{VAL}{VAL}, etc
\RequirePackage{forloop}        % \forloop command



%
% OBSŁUGA PARAMETRÓW
%

\SetupKeyvalOptions{family=KVP, prefix=KVP@} % KVP = Key Val Parameters

% deklaracje parametrów używanych przez tę klasę
\DeclareStringOption[polski]   {lang}          % pakiet językowy do włączenia, jeżli nie ⤸
chcemy podać pusty
\DeclareStringOption[11pt]     {fontSize}      % wielkość bazowej czcionki 8-20pt (klasa ⤸
extarticle)
\DeclareStringOption[a4paper]  {paperSize}     % rozmiar papieru (pakiet geometry)
\DeclareStringOption[portrait]{paperMode}      % orientacja papieru: portrait lub landscape
\DeclareStringOption[2.2cm]    {tmargin}       % górny margines
\DeclareStringOption[2.5cm]    {bmargin}       % dolny margines
\DeclareStringOption[2.2cm]    {lmargin}       % lewy margines
\DeclareStringOption[2.2cm]    {rmargin}       % prawy margines
\DeclareBoolOption   [true]     {twoside}       % tryb dwustronny z zaminą lewego i prawego ⤸
marginesy: true lub false
\DeclareBoolOption   [true]     {useMdSum}      % używanie sumy kontrolnej pliku: true lub ⤸
false; jeżeli wywołanie pdflatex
                                                % na to pozwala (opcja –shell-escape) suma ⤸
                                                zostanie automatycznie obliczona
\DeclareBoolOption   [true]     {attachSrc}     % załączanie wersji źródłowej do ⤸
generowanego pdf'u

\DeclareBoolOption   [true]     {sloppy}        % czy wolimy rozrzedzić linię niż wchodzić ⤸
na margines: true lub false
\DeclareStringOption[0pt]      {parindent}     % wielkość wcięcia akapitowego
\DeclareStringOption[1.17]     {stretch}       % wartość odstępu między liniowego (pakiet ⤸
setspace)
                                                % można podać także raw (nie zmieniana)
\DeclareStringOption[auto]     {baselineskip}  % wartość baselineskip, używana tylko gdy ⤸
stretch=raw można podać także:
                                                % raw (nie zmieniana) lub auto (1.17 pierwotnej)
\DeclareStringOption[auto]     {parskip}       % wartość parskip (dodatkowego odstępu ⤸
między akapitowego) można podać także:
                                                % raw (nie zmieniana) lub auto (1.23 ⤸
                                                baselineskip)
\DeclareBoolOption   [false]    {usePygments}   % czy chcemy używać  Pygments do kolorowania ⤸
kodu
```

```
\DeclareBoolOption    [true]     {noSectionNumbering}                    % wyłączenie ⤸
numerowania sekcji: true lub false
\DeclareComplementaryOption    {sectionNumbering}{noSectionNumbering}   % opcja komplementarna ⤸
do noSectionNumbering

\DeclareStringOption[]          {title}        % tytuł (dla właściwości pdf)
\DeclareStringOption[]          {authorName}   % autor (dla właściwości pdf)
\DeclareStringOption[]          {authorMail}   % adres e-mail autora (używany wraz z ⤸
authorName w właściwościach pdf)
\DeclareStringOption[]          {authorID}     % identyfikator autora (domyślnie authorMail) ⤸
używany w stopce

% przetwarzamy parametry
\ProcessKeyvalOptions*



%
% ŁADOWANIE KLASY BAZOWEJ i POZOSTAŁYCH PAKIETÓW
%

% klasa bazowa - article z rozszeronym zakresem wielkości bazowej czcionki (http://www.ctan.⤸
org/tex-archive/macros/latex/contrib/extsizes/)
\LoadClass[\KVP@fontSize]{extarticle}

% lokalizacja i pakiety podstawowe
\ifdefempty{\KVP@lang}{}{\RequirePackage{\KVP@lang}}
\RequirePackage{fontspec} % czionki dla LuaLaTex

% geometria strony i formatowania
\RequirePackage{geometry} % geometria strony - marginesy, ...
\RequirePackage{setspace} % odstępy między liniowe itp
\RequirePackage{titlesec} % kontrolowanie formatowania nagłówków sekcji
\RequirePackage{color} % kolory

% linki, interaktywne PDFy, ...
\RequirePackage{url} % łącza internetowe
\RequirePackage[unicode=true]{hyperref} % łącza internetowe, linki (także wewnętrzne), ⤸
właściwości PDF
\RequirePackage{navigator} % pozwala na załączanie plików do PDFu, tworzenie zakładek i ⤸
linków PDFowych

% naglowki, stopki, daty, ...
\RequirePackage{fancyhdr} % nagłówek i stopka
\RequirePackage[iso]{datetime} % aktualna data w iso
\RequirePackage{lastpage} % numer ostatniej strony wstawiany przez \pageref{LastPage}

% listy, pudełka i inne elementy składu
\RequirePackage[ampersand]{easylist} % obsługa elastycznych list numerowanych i ⤸
wypunktowywanych
\RequirePackage{alphalph} % obsługa literowego wypisywania duzych wartości liczników
\RequirePackage{pbox,varwidth} % vertical box z automatyczną detekcją minimalnej ⤸
szerokości
\RequirePackage{adjustbox} % pudełka przycinające oraz

% tabele i okolice
\RequirePackage{multirow,array,longtable,tabu} % lepsze, elastyczniejsze tabele
\RequirePackage{diagbox} % komórki dzielone na ukos
\RequirePackage{colortbl} % kolorowanie
```

```
\RequirePackage{hhline,dashrule} % zaawansowane linie

% czcionki
\RequirePackage{scalefnt,relsize} % skalowanie
\RequirePackage{soul,soulutf8} % highlighting, podkreślenia, przekreślenia, ...
\RequirePackage{ulem} % więcej podkreśleń, przekreśleń, ...
\RequirePackage{contour} % kolorowy kontur

% dodatkowe środowiska dla formatowań itp
\RequirePackage{ragged2e} % komenda \justifying
\RequirePackage{changepage} % srodowisko adjustwidth umozliwiajace wcinanie (także ujemne) ⤸
bloku

% ulepszona obsługa formatowania kodu maszynowego - minted i (wewnętrznie) fancyvrb
\ifKVP@usePygments \RequirePackage{minted} \else \RequirePackage[draft]{minted} \fi

% obrazki
\RequirePackage{graphics} % wlaczanie grafik
\RequirePackage{wrapfig} % rysunki (i tabele) opływane tekstem
\RequirePackage{overpic} % napisy na obrazkach
\RequirePackage{pdfpages} % włączanie wielustronnicowych PDFów
\RequirePackage{tikz}

% matematyka
\RequirePackage{amsmath,amssymb} % bardziej zaawansowane środowisko matematyczne, podwójne ⤸
literek trybu matematycznego
\RequirePackage{siunitx} % formatowanie liczb (m.in. wyrownanie względem separatora, ⤸
separator tysięcy, zaokrąglenia, jednostki)

% inne
\RequirePackage{todonotes} % komentarze, listy todo, ...
\RequirePackage{verbatim} % komentarze wieloliniowe



%
% OBSŁUGA INFORMACJI O PLIKU (w tym md5)
%

% w ramach tego pliku komendy itp definiowane są z użyciem kilku technik
% 1) \def... - rejestry i proste polecenia (także z argumentami)
% 2) \newcommand{...}{...} - bardziej złożone polecenia
% 3) \DeclareDocumentCommand - polecanie z zaawansowaną obsługą opcjonalnych argumentów

% komenda do pobierania zawartości pliku bez interpretowania jakichkolwiek znaków ⤸
specjalnych
\newcommand\CatchRawTextFile[2]{
 \CatchFileEdef{#1}{#2}{
  \catcode`\#=12 \catcode`\$=12 \catcode`\%=12 \catcode`\&=12 \catcode`\\=12
  \catcode`\^=12 \catcode`\_=12 \catcode`\{=12 \catcode`\}=12 \catcode`\~=12
 }
}

\ifKVP@useMdSum
% obliczenie md5 pliku źródłowego i wczytanie do rejestru \SrcFileInfo
 \immediate\write18{echo "`md5sum \jobname.tex` `pwd`/" > \jobname.md5}
 \IfFileExists{\jobname.md5}
  {\CatchRawTextFile{\SrcFileInfo}{\jobname.md5}}
  {\ClassError{office}
   {Can't find/generate md5 file (use h for help)}
```

```
   {You can:
    \MessageBreak\space\space - run pdflatex with --shell-escape to generate md5 sum
    \MessageBreak\space\space - run before pdflatex:
    \MessageBreak\space\space\space\space\space\space  echo '`md5sum "\jobname.tex"` `pwd`/' >⤸
    "\jobname.md5"
    \MessageBreak\space\space - add "useMdSum=false" to class paramters for disabel use of ⤸
    md5sum
   }
  }
 \immediate\write18{rm \jobname.md5}

% podział wczytanej informacji na pola
 \def\FileMDSum{\StrBefore[1]{\SrcFileInfo}{ }}
 \def\FileName {\StrBetween[1,2]{\SrcFileInfo}{ }{ }}
 \def\FilePath {\StrBehind[2]{\SrcFileInfo}{ }}
\else
 \def\FileMDSum{}
 \def\FileName {\jobname.tex}
 \def\FilePath {}
\fi

\ifKVP@attachSrc
\embeddedfile[TeX source file]{\jobname.tex}{\jobname.tex}
\fi



%
% KONFIGURACJA PAKIETÓW itp
%

% domyślna czcionka
\setmainfont[BoldFont=* Bold]{Linux Libertine O}\defaultfontfeatures{Ligatures=TeX}

% konfiguracja wstępna dla pakietu geometry
\geometry{%
   \KVP@paperSize,\KVP@paperMode,%
   tmargin=\KVP@tmargin,bmargin=\KVP@bmargin,lmargin=\KVP@lmargin,rmargin=\KVP@rmargin%
}
\ifKVP@twoside \geometry{twoside=true} \else \geometry{twoside=false} \fi

% przetwarzanie parametrów dla hyperref
\ifx\KVP@authorMail\empty
 \def\KVP@authorFullID{\KVP@authorName}
\else
 \def\KVP@authorFullID{\KVP@authorName~<\KVP@authorMail>}
\fi

% ustawienia dla hyperref (w tym parametry generowanego PDFu)
\hypersetup{%
  pdfauthor={\KVP@authorFullID},pdftitle={\KVP@title},%
  colorlinks=true,linkcolor=black,urlcolor=blue,unicode=true%
}

% domyslna (nadpisywana przez ustawienia nagłówka pliku) rozdzielczosc (ppi) wlaczanych ⤸
grafik
\pdfimageresolution=150

% preferencje do rozszerzen włączanych grafik
\DeclareGraphicsExtensions{.pdf,.mps,.png,.jpg}
```

```
% konfiguracja pakietu minted
\setminted{%
 breaklines=true, breakafter=-/.(), breakanywhere=true,
linenos=true
}



%
% KONFIGURACJA NAGŁÓWKA I STOPKI
%

% przetwarzanie parametrów dla stopki
\ifx\KVP@authorID\empty
 \def\KVP@authorID{\KVP@authorMail}
\fi

% nagłówki i stopki z wykożystaniem fancyhdr
\pagestyle{fancy}
\fancyhead{}\renewcommand{\headrulewidth}{0pt}
\fancyfoot{}\renewcommand{\footrulewidth}{0pt}
\fancyfoot[RO, LE]{
 \thepage~z~\pageref{LastPage}
}
\fancyfoot[LO, RE]{
 \vspace{-9pt} \fontshape{it}\fontsize{6pt}{0.7em}\selectfont \color[rgb]{.4,.4,.4}
 \parbox[b]{1.4cm}{
  user src:\\
  time md5:
}
 \pbox[b]{0.8\textwidth}{
  \KVP@authorID  \hspace{4pt}\hfill \FileName\\
  \today T\currenttime \hspace{4pt}\hfill \FileMDSum
 }
}



%
% USTAWIENIA AKAPITU itp
%

% przetwarzanie parametrów związanych z odstępami
\ifdefstring{\KVP@stretch} {raw}  {
 \ifdefstring{\KVP@baselineskip} {raw}  {\edef\KVP@baselineskip{\baselineskip}} {}
 \ifdefstring{\KVP@baselineskip} {auto} {\edef\KVP@baselineskip{1.17\baselineskip}} {}
} {
 \edef\KVP@baselineskip{\baselineskip}
 % ustawienie odstępów międzyliniowych z użyciem \setstretch
 \setstretch{\KVP@stretch}
}
\ifdefstring{\KVP@parskip}      {raw}  {\edef\KVP@parskip{\parskip}} {}
\ifdefstring{\KVP@parskip}      {auto} {\edef\KVP@parskip{1.23\baselineskip}} {}
\ifdefstring{\KVP@parindent}    {raw}  {\edef\KVP@parindent{\parindent}} {}
\ifdefstring{\KVP@parindent}    {auto} {\edef\KVP@parindent{0pt}} {}

% komenda ustawiająca rozmiary odstępów
% wywołanie komendy może być przydatne w parbox i środowisku minipage
% gdyż resetują one dla swojego wnętrza te wartości
```

```
\newcommand{\restoreSpacing}{
 % wcięcie akapitu
 \parindent=\KVP@parindent
% odległość między liniami
 \baselineskip=\KVP@baselineskip
%  dodatkowa odległość między akapitami
 \parskip=\KVP@parskip
}
% komendę wywołujemy też na początku dokumentu
\AtBeginDocument{\restoreSpacing}

% sloppy => wolimy rozrzedzenie linii od wchodzenia na margines
\ifKVP@sloppy
   \sloppy
\fi

% wyłączenie numerowania sekcji
\ifKVP@noSectionNumbering
\setcounter{secnumdepth}{0}
\fi



%
% USTAWIENIA LIST NUMEROWANYCH I WYPUNKTOWYWANYCH
%

% komenda pozwalająca na dodanie etykiety z nazwą (wyświetlaną przez \ref{ETYKIETA})
%  \label{ETYKIETA} dodaje etykietę której nazwą jest wartość rejestru \@currentlabel
%  \namedLabel{ETYKIETA}{NAZWA} dodaje etykietę której nazwą jest NAZWA
% komenda dodaje też etykietę PDFową wskazującą kokretne miejsce na stronie
\newcommand\namedLabel[2]{\begingroup%
 \def\@currentlabel{#2}%
 \def\@currentHref{page.\thepage}%
 \label{#1}\anchor{#1}%
\endgroup}

% obsługa konfigurowalnych etykiet do pozycji easylist
% 1) deklaracje liczników
\newcount\currEnumLevel
\newcount\maxEnumLevel
\newcount\minEnumLevel
% 2) komenda dodająca etykietę:
%   pierwszy argument: etykieta
%   drugi argument (opcjonalny): ilość poziomów numeracji do pominięcia od przodu (dla ⤸
referencji wenątrz listy numerowanej)
%   trzeci argument (opcjonalny): ilość poziomów numeracji do pominięcia od tyłu (⤸
nieistotne poziomy podpodpunktów)
%                              domyślnie pomija wszystkie poziomy listy wypunktowywanej
\DeclareDocumentCommand{\itemLabel}{ m O{0} o } {%
 \xdef\currRefLabel{}\scalebox{0.0001}{% sztuczka mająca na celu wyciszynie outputu ⤸
 produkowanego przez makra
   \currEnumLevel=1
   \minEnumLevel=#2
   \maxEnumLevel=\el@PreviousItem
   \IfValueTF{#3}{\advance\maxEnumLevel by -#3}{\DeterminateMaxEnumLevel}
   \RecurencyGetEnumLevel
 }%
 \namedLabel{#1}{\currRefLabel}%
}
```

```
% 3) komenda pomocnicza dla rekurencyjnego automatycznego pomijania poziomów listy ⤸
wypunktowywanej i nieuzywanych koncowych poziomow
\newcommand{\DeterminateMaxEnumLevel} {
 \ifnum\maxEnumLevel>\minEnumLevel
  \ifnum\maxEnumLevel=\csname Hide\the\maxEnumLevel\endcsname
   \advance\maxEnumLevel by -1
   \DeterminateMaxEnumLevel
  \else
   \ifnum\value{List\the\maxEnumLevel}=0
    \advance\maxEnumLevel by -1
    \DeterminateMaxEnumLevel
   \fi
  \fi
 \fi
}
% 4) komenda pomocnicza dla rekurencyjnego tworzenia nazwy dla etykiety
\newcommand{\RecurencyGetEnumLevel} {
 % if (minLev < currLev AND currLev <= maxLev AND value !=0)
 \ifnum\currEnumLevel>\minEnumLevel  \ifnum\currEnumLevel>\maxEnumLevel\else  \ifnum\value{⤸
 List\the\currEnumLevel}=0\else
  \xdef\currRefLabel{\currRefLabel%
  \csname el@NumberDenotation:\csname Numbers\the\currEnumLevel\endcsname\endcsname{⤸
  List\the\currEnumLevel}%
  \ifnum\currEnumLevel<\maxEnumLevel \csname Mark\the\currEnumLevel\endcsname \fi%
  }
 \fi  \fi  \fi
 % rekurencja
 \ifnum\currEnumLevel<\maxEnumLevel
  \advance\currEnumLevel by 1
  \RecurencyGetEnumLevel
 \fi
}

% redefinicje \alph i \Alph na funkcje z pakietu alphalph obslugujace numeracje powyzej z ⤸
jako: aa, ab, ac, ...
\let\alph\relax \def\alph#1{\alphalph{\value{#1}}}
\let\Alph\relax \def\Alph#1{\AlphAlph{\value{#1}}}

% definicje standardowych formatowan list
\def\initStdList{\NewList(
% separator pomiędzy numerami oraz separatory końcowe
 Mark=.,FinalMark=.,
% ustawienie odstępów między punktami
 Space=-0.35cm,Space*=-0.35cm,
% ustawienia wcięć
 Hang=true,Align=move,FinalSpace=0.5em,Progressive*=3ex,Margin1=1ex,
)}
\def\initTechList{\initStdList\ListProperties(
% formatowanie tekstu przypisane do 1go i 2go poziomu (używanych jako nagłówki)
 Style1=\Large\bf,Style2=\large\bf,
% na poziomie 5 uzywamy liter z separtorem końcowym ")"
 % i nie dołączamy numeracji wyższych poziomów
 Numbers5=l,Hide5=4,FinalMark5={)},Margin5=12ex,
% na poziomie 6 używamy wypunktowywania przy pomocy myslnika
 Hide6=6,Style6*=-- ,Margin6=15ex,
% na poziomie 7 uzywamy wypunktowywania z użyciem kropki
 Hide7=7,Style7*=$\bullet$ ,
)\xdef\subHeadLevel{3}}
\def\initTechSmallList{\initTechList\ListProperties(Style1=\large\bf,Style2=)⤸
\xdef\subHeadLevel{2}}
```

```
\def\initTechTinyList{\initTechList\ListProperties(Style2=,Margin5=9ex,Margin6=12ex)}
\def\initPointsList{\initStdList\ListProperties(
Hide1=1,Style1*=$\bullet$ ,Margin1=2ex,
Hide2=2,Style2*=-- ,Margin2=5ex,
)}

% komenda powodujaca pominiecie poziomu (ale po ustawieniu mu numeracji na 1)
\def\skipLevel#1{\ListProperties(Start#1=1)}

% komendy ustawiajace pogrubienie na zadanym poziomie
\def\isHead#1{\ListProperties(Style#1=\bf)}
\def\isNotHead#1{\ListProperties(Style#1=)}

% komenda wstawiajacapogrubiony item 2go lub 3go poziomu
\edef\AmpersandCatcode{\number\catcode`&}\catcode`&=\active
\def\headItem#1{
 \ifdefstring{\subHeadLevel}{2}{\isHead{2} &&  #1 \isNotHead{2}}{}
 \ifdefstring{\subHeadLevel}{3}{\isHead{3} &&& #1 \isNotHead{3}}{}
}
\catcode`&=\AmpersandCatcode



%
% USTAWIENIA TABEL
%

\newlength {\tabRowTmpHeight}
\newlength {\tabRowCurrHeight}
\newlength {\tabRowNewHeight}
\newlength {\tabRowForceHeight}
\newcounter{tabRowCount}
\newcounter{tabColCount}
\newsavebox{\tabCellBox}
\newsavebox{\tabCellVBox}

% force current row height
\newcommand{\forceRowHeight}[1]{\global\tabRowForceHeight=#1}

% @ TABLE ROW BEGIN
\newcolumntype{b}{%
 @{\global\tabRowNewHeight=0pt\global\tabRowForceHeight=0pt\stepcounter{tabRowCount}↩
 \setcounter{tabColCount}{1}%
   \ifcsname tabRowHeight@\AlphAlph{\value{tabRowCount}}\endcsname%
    \global\tabRowCurrHeight=\csname tabRowHeight@\AlphAlph{\value{tabRowCount}}\endcsname%
   \else%
    \global\tabRowCurrHeight=1pt%
   \fi%
  }%
}

% @ TABLE ROW END
\newcolumntype{e}{%
 @{\ifdim\tabRowForceHeight=0pt\else\tabRowNewHeight=\tabRowForceHeight\fi%
   \csdimgdef{tabRowHeight@\AlphAlph{\value{tabRowCount}}}{\tabRowNewHeight}%
   \immediate\write\@auxout{
     \string\newdimen\string\tabRowHeight@\AlphAlph{\value{tabRowCount}}
     \string\global\string\tabRowHeight@\AlphAlph{\value{tabRowCount}}=\the\tabRowNewHeight
   }%
  }%
```

```
}

% START TABLE CELL
% \tablePreCell{max width}{min width}{hmode}{vmode}
%   hmode = l (left) c (center) r (right) j (justify)
%   vmode = t (top) m (middle) b (bottom)
\DeclareDocumentCommand{\tablePreCell}{m m  m m}{%
  % start "varwidth" box in "lrbox" enviroment
  \begin{lrbox}{\tabCellBox}\begin{varwidth}[b]{#1}%
  % \\ oznacza nową linię tabeli
  \let\\\tabularnewline%
  \let\lb\linebreak
  % jeżeli okreslono minimalną szerokość
  \ifstrequal{#2}{}{}{%
    % dodajemy spację o takiej szerokości w ukrytym wierszu
    \settototalheight{\tabRowTmpHeight}{A}\hspace*{#2}\vspace{-\tabRowTmpHeight}\linebreak{}%
  }%
  % komenda wyrównanująca zawartość varwidth oraz proporcje odstępu poziomego przed i po ⤸
  varwidth
  \ifstrequal{#3}{l}{    \raggedright \gdef\tableCell@HPre{0}  \gdef\tableCell@HPost{1}}{%
   \ifstrequal{#3}{c}{   \centering    \gdef\tableCell@HPre{1}  \gdef\tableCell@HPost{1}}{%
    \ifstrequal{#3}{r}{  \raggedleft   \gdef\tableCell@HPre{1}  \gdef\tableCell@HPost{0}}{%
     \ifstrequal{#3}{j}{                \gdef\tableCell@HPre{0}  \gdef\tableCell@HPost{1}}{%
  }}}}%
  \gdef\tableCellVMode{#4}
  %
  % jeżeli na aktualnej kolumnie zdefiniowana jest wielowierszowość
  \ifcsname tabMultirowFirst@\the\value{tabColCount}\endcsname%
    % jeżeli jest to ostatni wiersz
    \ifnumcomp{\value{tabRowCount}}{=}{\csname tabMultirowLast@\the\value{tabColCount}⤸
    \endcsname}{%
      % wypisz rozciągniętą zawartość (w obecnej komórce) i zapomnij o niej
      \csname tabMultirowText@\the\value{tabColCount}\endcsname%
      \csundef{tabMultirowText@\the\value{tabColCount}}%
    }{}%
  \fi%
}

% FINISH TABLE CELL
\DeclareDocumentCommand{\tablePostCell}{ O{1} }{%
  \@finalstrut\@arstrutbox\end{varwidth}\end{lrbox}%
  % pobieramy wysokość aktualnej komórki
  \settototalheight{\tabRowTmpHeight}{\usebox{\tabCellBox}}%
  \dimdef{\multirowHeight}{0pt}%
  %
  % jeżeli na aktualnej kolumnie zdefiniowana jest wielowierszowość
  \ifcsname tabMultirowFirst@\the\value{tabColCount}\endcsname%
    \numdef{\lastRow}{\csname tabMultirowLast@\the\value{tabColCount}\endcsname}%
    % jeżeli jest to ostatni wiersz
    \ifnumcomp{\value{tabRowCount}}{=}{\lastRow}{%
      \newcounter{row}%
      \forloop{row}{\csname tabMultirowFirst@\the\value{tabColCount}\endcsname}{\value{row} < ⤸
      \lastRow}{%
        \dimdef{\multirowHeight}{\multirowHeight +  \csname tabRowHeight@\AlphAlph{\value{row}}⤸
        \endcsname +  5pt}% FIXME
      }%
      % usuń definicje wielowierszowści
      \csundef{tabMultirowFirst@\the\value{tabColCount}}%
      \csundef{tabMultirowLast@\the\value{tabColCount}}%
      \csundef{tabMultirowForceHeight@\the\value{tabColCount}}%
```

```
    }{}%
  \fi%
  %
  % ustalamy maksymalną wysokość komórki w wierszu
  \setlength{\tabRowNewHeight}{\maxof{\tabRowNewHeight}{\tabRowTmpHeight - \multirowHeight}}%
  % to musi być globalna długość ...
  \global\tabRowNewHeight=\tabRowNewHeight%
  % ustalamy wysokość wypełnień której mamy użyć w obecnym przebiegu
  \typeout{CELL(\the\value{tabRowCount},\the\value{tabColCount}): max(\the\tabRowNewHeight, ⤸
  \the\tabRowCurrHeight) + \multirowHeight - \the\tabRowTmpHeight}%
  \setlength{\tabRowTmpHeight}{\maxof{\tabRowNewHeight}{\tabRowCurrHeight} + \multirowHeight -⤸
  \tabRowTmpHeight}%
  % jeżeli mamy wyśrodkować w pionie to potrzebujemy 1/2 \tabRowTmpHeight
  \ifdefstring{\tableCellVMode}{m}{\setlength{\tabRowTmpHeight}{0.5\tabRowTmpHeight}}{}%
  %
  % przygotowujemy vbox z przygotowanym wcześniej varwidth i dodatkowym wypełnieniem ⤸
  pionowym
  \begin{lrbox}{\tabCellVBox}\vbox{%
    % poprzedzony odstępem pionowym w trybach b oraz c (czyli nie t) + zawsze 4pt
    \ifdefstring{\tableCellVMode}{t}{}{\vspace*{\tabRowTmpHeight}}\vspace*{4pt}% FIXME
    \hbox{\usebox{\tabCellBox}}%
    % zamknięty odstępem pionowym w trybach t oraz c (czyli nie c) + zawsze -4pt
    % \vspace*{0pt} ma efekt i jest potrzebne aby było równo
    \ifdefstring{\tableCellVMode}{b}{\vspace*{0pt}}{\vspace*{\tabRowTmpHeight}}\vspace*{-4pt}⤸
    % FIXME
  }\end{lrbox}%
  % jeżeli wypisujemy multirow to modyfikujemy wysokość zapamiętanego vboxu
  \ifdim\multirowHeight=0pt\else \ht\tabCellVBox=0pt \dp\tabCellVBox=0pt \fi
  %
  % wypełniający odstęp po lewej (wyrównanie varwidth)
  \hspace{\stretch{\tableCell@HPre}}%
  % wypisujemy przygotowany vbox
  \usebox{\tabCellVBox}
  % wypełniający odstęp po pawej (wyrównanie varwidth)
  \hspace{\stretch{\tableCell@HPost}}%
  %
  % zwiększamy licznik kolumn
  \addtocounter{tabColCount}{#1}%
}

% potrzebujemy kolumny tupu j aby w Z móc używać #3 jako typu kolumny bazowej
\newcolumntype{j}{l}

% Z{max width}[min width]{hmode}{col num}{vmode}{rown num}
%   hmode = l (left) c (center) r (right) j (justify)
%   vmode = t (top) m (middle) b (bottom)
% używamy hmode jako typu kolumny bazowej bo inaczej w multicolumn może dziwnie się ⤸
wyrównywać
\newcolumntype{Z}[6]{>{\tablePreCell{#1}{#2}{#3}{#5}}#3<{\tablePostCell[#4]}}



\newcolumntype{L}[3]{Z{#1}{#2}{l}{1}{#3}{1}}
\newcolumntype{C}[3]{Z{#1}{#2}{c}{1}{#3}{1}}
\newcolumntype{R}[3]{Z{#1}{#2}{r}{1}{#3}{1}}
\newcolumntype{J}[3]{Z{#1}{#2}{j}{1}{#3}{1}}

% \tableFormatedCell{max width}[min width]{hmode}{vmode}
%   hmode = l (left) c (center) r (right) j (justify)
%   vmode = t (top) m (middle) b (bottom)
```

```
% używamy hmode jako typu kolumny bazowej bo inaczej w multicolumn może dziwnie się ⤸
wyrównywać
\DeclareDocumentCommand{\tableFormatedCell}{m O{} m m}{
  \@finalstrut\@arstrutbox\end{varwidth}\end{lrbox}%
  \tablePreCell{#1}{#2}{#3}{#4}%
}

% \mymulticolumn{col num}{max width}{min width}{hmode}{vmode}{right sep}{text}
%   hmode = l (left) c (center) r (right) j (justify)
%   vmode = t (top) m (middle) b (bottom)
\newcommand{\mymulticolumn}[7]{\multicolumn{#1}{Z{#2}{#3}{#4}{#1}{#5}{1}#6}{#7}}

\DeclareDocumentCommand{\setmultirow}{m O{0pt} m}{%
  \csnumgdef{tabMultirowFirst@\the\value{tabColCount}}{\value{tabRowCount}}
  \csnumgdef{tabMultirowLast@\the\value{tabColCount}}{\value{tabRowCount} + #1 - 1}
  \csdimgdef{tabMultirowForceHeight@\the\value{tabColCount}}{#2}
  \csgdef{tabMultirowText@\the\value{tabColCount}}{#3}
}

% patch \HH@loop macro from hhline to add !{} syntax
\AtBeginDocument{
  \patchcmd{\HH@loop}{\PackageWarning{hhline}}{%
   \ifx\@tempb!\def\next#1#2{%
       \gdef\HH@height{\dimen\thr@@}%
       \if@firstamp\@firstampfalse\else\HH@add{&\omit}\fi
       \HH@add
          {#2}%
       \HH@let!}\else
   \PackageWarning{hhline}
  }{}{\ClassWarning{Error patch HH@loop}}
  \patchcmd{\HH@loop}{\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi}{\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi}{}{⤸
  \ClassWarning{Error patch HH@loop}}
}

% kolumna wstawiająca pionowe obramowanie z użyciem hdashrule
\newcolumntype{I}[3]{!{%
  \setlength{\tabRowTmpHeight}{\tabRowCurrHeight + 5pt}% FIXME
  \begin{lrbox}{\tabCellVBox}%
  \rotatebox{90}{#1\hdashrule{\tabRowTmpHeight}{#2}{#3}}%
  \end{lrbox}%
  \ht\tabCellVBox=0pt%
  \dp\tabCellVBox=0pt%
  \usebox{\tabCellVBox}%
}}



%
% DODATKOWE KOMENDY
%

% definicja nowych komend dla trybu matematycznego:
%   ułamek a/b z różnicą wysokości zapisu a i b
\newcommand\minifrac[2]{\raisebox{.3ex}{$#1$}/\raisebox{-.6ex}{$#2$}}
%   kreska używana do pokazania "w punkcie" (np. przy pochodnej)
\newcommand\inpoint[1]{\hspace{.4ex}\raisebox{-.55ex}{\scalebox{0.7}{$\bigg|_{#1}$}}\hspace{-.⤸
8ex}}

% komenda pozwalająca na modyfikację geomerii strony w treści dokumentu - w odróznieniu ⤸
od newgeometry:
```

```
% 1. zachowuje dotychczasowe ustawienia pozwalajac je modyfikować
% 2. pozwala także na zmianę orintacji i wielkości papieru (działa tylko dla pdflatex ⤸
!!!)
\newcommand{\forceNewPageGeometry}[1]{%
  \clearpage%
  \Gm@clean\setkeys{Gm}{#1}\Gm@process%
  \eject \pdfpagewidth=\paperwidth \pdfpageheight=\paperheight%
  \Gm@changelayout%
  \headwidth=\textwidth%
}

\newcommand{\highlight}[1]{%
 \immediate\write18{highlight -f -O latex -o "\jobname.tmp" "#1"}%
 \ifdef{\hlstd}{}{%
  \immediate\write18{echo "" | highlight --out-format=latex -S tex > /dev/null}%
  \input{highlight.sty}%
 }%
 \input{\jobname.tmp}
}

% komenda wstawiająca załącznika
\DeclareDocumentCommand{\zalacznik}{m O{} m O{11cm}}{
 %\protected@write\@auxout{}{\string\@writefile{loa}{& #1 -- #3}}
 \addtocontents{loa}{& #1 -- #3}
 \includepdf[
  fitpaper=true,
  pagecommand={
   %\setlength{\headheight}{110pt}\fancyhead[C]{}\renewcommand{\headrulewidth}{0.5pt}⤸
   \pagestyle{fancy}
   \thispagestyle{empty}
   \adjustbox{margin=1em,bgcolor={rgb}{0.9,0.9,0.9}}{\pbox{#4}{\small Załącznik: #1 -- #3}}
  },
  #2
 ]{zalaczniki/#1.pdf}
}

% komenda wstawiająca spis załączników wstawianych z użyciem \zalacznik
\DeclareDocumentCommand{\spisZalacznikow}{}{
 \pagebreak\textbf{\large Wykaz załączników:}
 \begin{easylist}\initPointsList
 \@starttoc{loa}
 \end{easylist}
 \AtEndDocument{\immediate\write\@auxout{\string\@writefile{loa}{\string\ListProperties(⤸
Style1*={}) &}}} % dirty-hack
}
```

OFFICE-DEMO.TEX

```
% określamy klasę dokumentu (może być jakaś standardowa lub nasza własna jak w tym ⤸
wypadku)
% i przekazujemy jej parametry
\documentclass[%
 useMdSum=true,lmargin=3cm,twoside=false,noSectionNumbering=true,%
 authorName={Robert Paciorek},authorMail={robert@opcode.eu.org},title={demo},%
 parindent=10pt,usePygments=true,%
 %stretch=raw,baselineskip=22pt%
]{office}
```

```
% możemy podać także inne ustawienia pakietu geometry służącego do określania układu ⤸
strony
% więcej szczegółów: http://mirrors.ctan.org/macros/latex/contrib/geometry/geometry.pdf
\geometry{verbose} % showframe - pozwala na pokazanie obramowań strony, marginesów, etc

% możemy podać inne / dodatkowe ustawienia pakietu hyperref służącego do konfiguracji ⤸
interaktywnych odnośników, usatwień pdf, itd
% więcej szczegółów: http://mirrors.ctan.org/macros/latex/contrib/hyperref/doc/manual.pdf
\hypersetup{linkcolor=red}

% przydatne pakiety
\usepackage{fancypar} % indywidualne formatowanie linii w akapicie
  \newcommand*\fancyparDemoStyle[1]{%
    \bgroup
    \textwidth=0.5\textwidth
    \hsize=\textwidth
    \linewidth=\textwidth

    \renewcommand\FancyPreFormat{
      \setcounter{fancycount}{0}
    }
    \renewcommand\FancyFormat{%
      \stepcounter{fancycount}
      \ifnum\value{fancycount}=1%
        \textcolor{red}{\box\linebox}%
      \else%
        \box\linebox%
      \fi%
    }
    \par\medskip%
    \hspace{2cm}\vbox{\noindent#1\par\AddFancyFormat}%
    \par\medskip%
    \egroup
  }
\usepackage{fancytabs} % zakładki na marginesie
\usepackage{spreadtab} % arkusz kalkulacyjny
\usepackage{pgfplots} % wykresy
\usepackage{pdfcomment,attachfile,marginnote} % komentarze pdf-owe


% skrót na uzyskiwanie backslasha
\newcommand{\bs}[1]{\textbackslash#1}

% definiujemy środowisko InlineVerbatim, będące środowiskiem Verbatim z zmniejszonymi ⤸
ostępami przed i po
\newenvironment{InlineVerbatim}{
 \VerbatimEnvironment
 \vspace{-\partopsep}\vspace{-\parskip}
 \begin{Verbatim}
}{
 \end{Verbatim}
 \vspace{-\parskip}
}
% oraz InlineVerbatim*, będące środowiskiem Verbatim z zmniejszonymi ostępami przed
\newenvironment{InlineVerbatim*}{
 \VerbatimEnvironment
 \vspace{-\partopsep}\vspace{-\parskip}
 \begin{Verbatim}
}{
 \end{Verbatim}
```

```
}

% definiujemy komendy do wypisywania przykładu (po prawej) z kodem (po lewej)
\newcommand{\PrintCode}[2]{
 \colorbox[rgb]{.9,.9,.9}{\parbox[c]{#1}{%
  \vspace{-\topsep}\vspace{-\partopsep}\vspace{-\parskip}%
  \VerbatimInput[numbers=left,numbersep=4pt,fontsize=\footnotesize]{#2}%
  \vspace{-\topsep}\vspace{-\partopsep}%
 }}%
}
\newcommand{\PrintExample}[2]{
 \parbox[c]{#1}{\centering%
  \input{#2}%
 }%
}
\newcommand{\PrintCodeAndExampleA}[1]{
 \nopagebreak

\hspace{0.015\textwidth}\PrintCode{0.6\textwidth}{#1}%
 \hspace{0.03\textwidth}\fbox{\PrintExample{0.3\textwidth}{#1}}%
}
\newcommand{\PrintCodeAndExampleB}[1]{
 \begin{adjustwidth}{.05\textwidth}{.05\textwidth}
  \let\FancyVerbFormatLineOrg\FancyVerbFormatLine
  \def\FancyVerbFormatLine##1{%
   \colorbox[rgb]{.9,.9,.9}{\hbox to .975\linewidth {##1}}%
  }%
  \VerbatimInput[numbers=left,numbersep=4pt,fontsize=\footnotesize]{#1}
  \let\FancyVerbFormatLine\FancyVerbFormatLineOrg%
 \end{adjustwidth}\vspace{-0.5cm}%
 \begin{center}%
  \input{#1}
 \end{center}
}

% definiujemy środowiska Example i ExampleVertical
\makeatletter % powoduje że możemy używać @ w nazwach makr
 \@namedef{Example}{\FV@Environment{}{VerbatimOut}{\jobname.tmp}}
 \@namedef{endExample}{\@nameuse{FVE@VerbatimOut}\PrintCodeAndExampleA{\jobname.tmp}}
 \@namedef{ExampleVertical}{\FV@Environment{}{VerbatimOut}{\jobname.tmp}}
 \@namedef{endExampleVertical}{\@nameuse{FVE@VerbatimOut}\PrintCodeAndExampleB{\jobname.tmp}}
\makeatother % przywracamy standardowe znaczenie @

% lipsum służy do wypisywania zapychającego tekstu ...
\usepackage{lipsum}


%% Modyfikujemy parametry wpływajce na skład i łamanie *** TP ***
%\overfullrule = 3mm % zaznacz linijki z~nadmiarami czarn sztabk *** TP
\exhyphenpenalty = 999999 % słowa złożone nie przeno na łczniku *** TP
\righthyphenmin = 3 % nie dziel xx-, ale xxx- *** TP
\widowpenalty = 999999 % ostatnia linia akapitu nie zacznie strony *** TP
\clubpenalty = 9950 %% pierwsza linia akapitu wyjtkowo może kończyć stronę
\hfuzz = 3pt %% znaki mog sterczeć na margines do wielkoci 3pt *** TP
\tolerance 600 %% zwiększamy tolerację ***
\pretolerance 300 %% oraz pretolerancję ***
\doublehyphendemerits = 99999 %% zniechęcamy TeX-a do dzielenia wyrazów *** TP

\definecolor{hl_yellow}{rgb}{1,1,0}
\newcommand{\hlcb}[2][hl_yellow]{\sethlcolor{#1}\hl{\textbf{#2}}}
```

```
\begin{document}


% spis treści
% możemy go dawać od przodu - LaTeX radzi sobie z offsetem w numeracji przez niego ⤸
wprowadzanym
% jeżeli jest dłuższy niż 1 strona (wymaga to tylko dodatkowej generacji dokumentu)
\addcontentsline{toc}{section}{Spis treści} % cheby aby spis tereści był w spisie treści ⤸
jako sekcja
\def\contentsname{Spis treści} % chemy aby spis treści nosił nagłówek ”Spis treści”
                               % (nagłówek ten będzie formatowany jako nagłówk 1go ⤸
                               poziomu - sekcji)
\setcounter{tocdepth}{2} % chemy aby w spisie treści były dwa najwyższe poziomy (section i ⤸
subsection)
{
\parskip=0pt
\tableofcontents
}
\clearpage


% czasowo włączamy numerację ...
\setcounter{secnumdepth}{3}
\section{Formatowanie nagłówków}

Standardowo w article na którym bazujemy dostępne są 3 poziomy nagłówków typowo z ⤸
włączoną ich numeracją:
\subsection{poziom 2 - subsection}
\subsubsection{poziom 3 - subsubsection}


Na fakt ich numerowania lub nie możemy wpływać np. poprzez ustawienie:\\ \verb#\setcounter{⤸
secnumdepth}{2}# włączamy numerację do 2go poziomu włącznie. Do bardziej zaawansowanego ⤸
manipulowania numeracją można użyć redefiniowania komend \verb#\thesection# ⤸
\verb#\thesubsection# itd. np.:

\begin{Example}
\renewcommand{\thesubsection}{
  \thesection-\arabic{subsection})
}
\renewcommand{\thesubsubsection}{
  \thesection-\arabic{subsection}-\arabic{subsubsection})
}
\subsection{subsection}
\subsubsection{subsubsection}
\end{Example}

% wracamy do domyślnego ustawienia dokumentu uzyskanego przez noSectionNumbering=true
\setcounter{secnumdepth}{0}


Możemy także szczegułowo wpływać na formatowanie poszczególnych nagłówków, jak ⤸
również odstępy występujące przed i za nagłówkami z użyciem komend:
\begin{InlineVerbatim}
 \titleformat{ modyfikowany nagłówek }
    { formatowanie }{ numeracja }{ odległość między numeracją a tekstem }
    { kod do dodania przed }{ kod do dodania po }
\end{InlineVerbatim}
```

```
oraz \verb#\titlespacing# z pakietu \href{http://mirrors.ctan.⤸
org/macros/latex/contrib/titlesec/titlesec.pdf}{titlesec}.
Dzięki temu teraz nagłówki 1-go poziomu są wyśrodkowane i używają trochę większej ⤸
czcionki, a nagłówki pozimu 2 mają zmniejszone odstępy.

% ustawiamy aby nagłówki sekcji były wyśrodkowywane
\titleformat{\section}{\LARGE\bfseries\centering}{\thesection}{1em}{\pagebreak[3]}{⤸
\nopagebreak}
% jeżeli chemy zmieniać samo formatowanie możemy użyć uproszczonej wersji:
%  \titleformat*{\section}{\LARGE\bfseries\centering}
%
% podobny efekt można uzyskać stosując:
%  \renewcommand*\section{\@startsection{section}{1}{-3.0ex}{3.5ex}{3.5ex}{⤸
\centering\LARGE\bfseries}}
% lub:
%  \DeclareSectioningCommand{section}{1} {24pt} {\centering\LARGE\bfseries\HeadingText} {10pt}
%
% zmniejszamy odległości wokół nagłówków sub-sekcji
\titlespacing{\subsection}{0pt}{-0.2cm}{-0.5cm}



\section{Listy numerowane i wypunktowywane}

\begin{easylist}\initTechList
& są różne sposoby wyliczania \label{AA}
&& pakiet enumerate
&&& pakiet ten umożliwia m.in.:
&&&& defionwanie sposobu wyliczania (liczby, litery, ...)
&&&& defionwanie separatora występującego po liczniku
&& pakiet enumitem
&&& umożliwia on m.in. definiowanie odstępów, margnesów itp
&&& nie ejst stosowany w tym dokumencie
&& pakiet easylist
&&& ta lista jest utworzona właśnie z jego użyciem\\oraz zdefiniowanych w office.⤸
cls\\formatowań list i przydatnych udogodnień \\ \\ (jak widać możemy łamać linie)
&&& i nie tylko ...

&& obecny punkt to: \arabic{List1}.\arabic{List2}.\arabic{List3}.\arabic{List4}.\alph{List5}.
\isHead{3} &&& naglowek na poziomie 3 \isNotHead{3}
&&&& $2 \over x$ - matematyka też działa &&&&& obecny punkt to:
% to czy piszemy w nowych liniach czy nie jest bez znaczenia
 \arabic{List1}.\arabic{List2}.\arabic{List3}.\arabic{List4}.\alph{List5}.
&&& podpunkt poziomu 3 nie będący nagłówkiem
\headItem{i kolejny nagłówek, ale inaczej uzyskany}
&&& jesteśmy w \ref{AA}
&&&& podpunkt poziomu 4 \itemLabel{BB}
&&&&&  i następny w \ref{BB}
&&&&&& i jeszcze następny \itemLabel{CC}
&&&&&&& i jeszcze następny w podpodpunkcie \ref{CC}
&&&&& wracamy \label{D1} \itemLabel{D2} do literek  \ref{D1} vs \ref{D2}
\ListProperties(Start5=25)
&&&&& przeskoczyliśmy parę numerków do \ref{E2} w \ref{E3} (\ref{E1})
      \itemLabel{E1} \itemLabel{E2}[4] \itemLabel{E3}[0][1]
&&&&& aby pokazać że dzięki użyciu alphalph
&&&&& możemy numerować powyżej "z" (26ciu)
& wracamy do poziomu 1
\skipLevel{2}
&&& przeskakujemy jeden
&&&&&& i od razu punktujemy \itemLabel{F2} referencja to \ref{F2}
```

```
\end{easylist}



\section{Formatowanie tekstu}

% jest to lista wyliczana z trochę innym formatowaniem niż powyższa
% - drugi poziom nie jest "nagłówkowy"
% - mamy też do dyspozycji trzeci
% - literki są tak naprawdę na piątym poziomie, ale na to nie wyglądają
\begin{easylist}\initTechTinyList
& Możemy korzystać z komend ustawijących:
&& dla aktualnego bloku ujętego w \{ \}
&&&&& wielkość:
{\tiny \bs{tiny}}
 {\scriptsize \bs{scriptsize}}
 {\footnotesize \bs{footnotesize}}
 {\small \bs{small}}
 {\normalsize \bs{normalsize}}
 {\large \bs{large}}
 {\Large \bs{Large}}
 {\LARGE \bs{LARGE}}
 {\huge \bs{huge}}
 {\Huge \bs{Huge}}
&&&&& formatowanie:
{\bfseries \bs{bfseries}}
 {\itshape \bs{itshape}}
 {\normalfont \bs{normalfont}}
 {\scshape \bs{scshape}}
 ...
&&&&& krój / typ czcionki:
{\ttfamily \bs{rmfamily}}
 {\sffamily \bs{sffamily}}
 {\rmfamily \bs{rmfamily}}
&&&&& kolor:
{\color[rgb]{0,1,0} \bs color[rgb]\{0,1,0\}}
&& dla swojego argumentu:
&&&&& formatowanie:
\textbf{\bs textbf}
 \textit{\bs textit}
 \textnormal{\bs textnormal}
 \textsc{\bs textsc} ...
&&&&& krój / typ czcionki:
\texttt{\bs texttt}
 \textsf{\bs textsf}
 \textrm{\bs textrm}
&&&&& kolor:
\textcolor[rgb]{0,1,1}{ \bs textcolor[rgb]\{0,1,1\}}
&&&&& efekty - pakiet soul (kolory można swobodnie zmieniać):
\so{rozrzedzony}
 \caps{Samll and BIG Capitals}
 \setstcolor{red} \st{przekreslony na czerwono}
 \setulcolor{green} \ul{podkreślony na zielono}
 \sethlcolor{yellow} \hl{podswietlony na żółto}
&&&&& efekty - pakiet ulem:
\sout{\bs sout}
 \xout{\bs xout}
 \uline{\bs uline}
 \uuline{\bs uuline}
 \uwave{\bs uwave}
```

```
 \dashuline{\bs dashuline}
  \dotuline{\bs dotuline}
\end{easylist}

Krój maszynowy możemy też uzyskać przez komendę \bs verb, środowisko "verbatim" i im ⤸
podobne. W odróżnieniu od wyżej prezentowanych poleceń wprowadzających krój maszynowy ⤸
wyłączają one interpretowanie jakichkolwiek poleceń Tex poza znacznikiem kończącym:
\vspace{-0.6cm} % zmiejszamy odległość poprzedzającą środowisko verbatim
                % może nie jest to najbardziej eleganckie rozwiązanie,
                % ale dobrze demonstruje użycie \vspace którym możemy regulować
                % (wstawiać dodatkowe dodatnie lub ujemne) odstępy pionowe
\begin{Verbatim}[formatcom=\color{red}]
teskt maszynowy
z  dziwnymi $&_#&    znaczkami, układem   spacji itd

analogicznie działa inline-owy \verb#tekst#
gdzie # może być zastapiony dowlolnym znaczkiem
(ale nie wolno mu występować w tekście)
\end{Verbatim}
\noindent % zaczynamy bez wcięcia
Zastosowane środowisko Verbatim z pakietu \href{http://www.ctan.org/pkg/fancyvrb}{fancyvrb} ⤸
oprócz zaprezentowanej zmiany kolororu, umożliwia także m.in. zmianę czcionki (fontfamily,⤸
fontshape, fontsize, ...), numerowanie linii, konfigurację etykiet, obramowania. W ⤸
odróżnieniu od klasycznego verbatim nie można zaczynać tekstu w linii w której jest ⤸
\verb#\begin{Verbatim}#.

Możliwe też jest wstawianie formatowanego kodu:
Pakiet \href{http://mirrors.ctan.org/macros/latex/contrib/minted/minted.pdf}{minted} (⤸
używający wewnętrznie fancyvrb) umożliwia wstawianie fragmentu kodu "inline" (\mintinline{⤸
tex}{\mintinline{tex}{kod}}), wstawianie formatowanego bloku kodu poprzez środowisko minted ⤸
oraz wstawianie kodu z pliku źródłowego (z możliwością określenia np. zakresu linii ⤸
które mają być użyte) poprzez \mintinline{tex}{\inputminted}. Pakiet posiada także ⤸
bogate opcje konfiguracyjne związane z wyświetlaniem kodu. Potrafi m.in. łamać linię w ⤸
środowisku wypisywanym czcionką maszynową oraz kolorować kod w oparciu o \href{http:⤸
//pygments.org/}{pygments}.
\begin{minted}{c}
int main() {
 int a=5;
printf("%d\n", a);
}
\end{minted}

Jeżeli potrzebujemy dużej czcionki możemy użyć skalowania:
\begin{Example}
{\scalefont{2.5} wiekszy 2.5 raza} normalny
{\relsize{2.5} o 2.5 stopnia wiekszy}
\end{Example}

Ponaddto możemy także definiować czcionki z wykożystaniem polecania font:
\begin{Example}
 \font\duzypochyly=plbxti10 scaled 3300
 {\duzypochyly DEMO demo}
\end{Example}

Inną metodą na swobodne określanie kroju czcionki jest użycie niskopoziomowych komend ⤸
określających poszczególne parametry czcionki np:
\begin{Example}
{
 \fontencoding{T1}      % kodowanie
 \fontfamily{antt}      % kroj
```

```
  \fontseries{b}              % formatowanie
  \fontshape{it}              % formatowanie
  \fontsize{32pt}{1.7em}  % rozmiar
  % ustawiamy okreslona wczesniej czionke:
  \selectfont DEMO demo
}
\end{Example}

Kolejne sposoby deklarowania kroju czcionku daje nam pakiet fontspec, który w połączeniu z ⤸
Lua\LaTeX{} pozwala na używanie czcionek OpenType. Podstawowymi poleceniami używanymi do ⤸
określania czcionki z użyciem tego pakietu są \bs fontspec określający krój czcionki (⤸
poprzez jego nazwę lub plik) oraz \bs addfontfeatures określajacy jej atrybuty (mozemy je ⤸
także podawać w opcjonalnym argumencie \bs fontspec.

\begin{Example}
{ \fontspec{Latin Modern Roman}
  In 1842, 999 pe sailed 97 miles
  \addfontfeatures{
     Numbers={Proportional,OldStyle},Color=888888}
  In 1842, 999 pe sailed 97 miles
}
\end{Example}

% zmieniamy formatowanie linii dla środowiska Verbatim (używanego przez Example)
% komenda ta może posłużyć do bardziej zaawansowanych operacji formatowania nawet każdej ⤸
z linii z osobna
\let\FancyVerbFormatLineOrg\FancyVerbFormatLine
\def\FancyVerbFormatLine#1{%
\fontspec{DejaVu Sans Mono}\scriptsize% zmieniamy czcionkę na obsługującą dzwine znaczki
\ifnum\value{FancyVerbLine}=3% oraz jako bonus dla 3ciej linii ustawiamy kolor tła na ⤸
czerwony
  \colorbox[rgb]{1,0,0}{\hbox to .975\linewidth {#1}}%
\else#1\fi%
}
\begin{Example}
{ \fontspec{DejaVu Sans} \small
  Możemy także używać różnych znaczków Unicode
   wprost w pliku  ale to my
  odpowiadamy za wybór czcionki która je obsługuje
}
\end{Example}
% przywracamy domyślne
\let\FancyVerbFormatLine\FancyVerbFormatLineOrg


%deklarowanie kroju - przypisywanie komendy typu \mojKroj do czcionki - \newfontfamily (⤸
pozwala na stosowanie modyfikatorów typu \textbf) \newfontface (nie pozwala na zmienianie ⤸
wariantu czcionki)


%\setmainfont{TeX Gyre Pagella}
%\setsansfont{TeX Gyre Heros}%[Scale=MatchLowercase]
%\setmonofont{Inconsolata}%[Scale=MatchLowercase]

% wcięty na 1cm od lewej i 4cm od prawej blok
\begin{adjustwidth}{1cm}{4cm}
  \textbf{poczatek wciętego akapitu}\\
Ten wcięty (z obu stron) akapit został uzyskany z użyciem środowiska adjustwidth,
pozwalającego na tworzenie takich fragmentów tekstu z zachowaniem podziału na strony.
Poniżej w ramach Verbatim wypisane są informacje o szerokości tekstu
```

```
(jak widać w ramach Verbatim) da się używać komend \TeX{}'owych)
% \the powoduje wypisanie występującego po nim licznika, odległości itp
  % jeżeli chemy ich używać jak wartości numerycznych to nie korzystamy z \the
  % jeżeli wypisujemy w tekście a by nie stracić spacji po takim wypisaniu należy użyć {}⤸
 - np. tekst \the\hsize{} teskt
 Poniżej mamy wypełniacz stworzony z użyciem lipsum
i wymuszania nowego akapitu poprzez \verb#\par# (a nie pustej linii).

\lipsum*[4] \par \lipsum*[5]
 \\ \textbf{koniec wciętego akapitu}
\end{adjustwidth}

% akapit z niezalezniе formatowanymi liniami - patrz definicja \fancyparDemoStyle w ⤸
nagłówku
\fancyparDemoStyle{Pakiet \href{http://mirrors.ctan.⤸
org/macros/latex/contrib/fancypar/fancypar.pdf}{fancypar}
jest wygodnym interfejsem do modyfikowania formatowania akapitu z niezależnym formatowaniem ⤸
każdej linii
(np. tutaj ustawiliśmy że pierwsza ma być w kolorze czerwonym). Pozwala też na uzyskanie ⤸
podobnego efektu jak adjustwidth}

Z kolei pakiet \href{http://mirrors.ctan.org/macros/latex/contrib/contour/contour.pdf}{⤸
contour} pozwala na uzyskanie \contourlength{0.04em}\contour[1]{red}{konturowego} ⤸
\contourlength{0.07em}\contour[3]{red}{kolorowego} \contourlength{0.03em}\contour[10]{red}{⤸
tła liter}

W ramach listy wypunktowywanej kilka informacji o formatowaniu akapitu (odstępach ⤸
międzyliniowych itp):
\vspace{-0.66\parskip}\begin{easylist}\initPointsList
& odstępy międzyliniowe:
&& baselineskip: \the\baselineskip
&& baselinestretch: \baselinestretch
& dodatkowy odstęp między akapitowy
&& parskip: \the\parskip
& inne: parindent=\the\parindent{} lineskip: \the\lineskip{}
\end{easylist}



\section{Pudełka, grafika i geometria strony}

\hspace{\stretch{2}}
  % możemy wstawić obrazek skalując go (względem jego rozmiaru wynikającego z jego ⤸
 rozdzielczości)
 \parbox[c]{0.4\textwidth}{\centering
  \includegraphics[scale=0.44]{/usr/share/texlive/texmf-dist/tex/latex/mwe/example-image}
  % gdy nie podajemy rozszerzenia latex sam wybierze optymalny format zgodnie z sugestią ⤸
  podaną w \DeclareGraphicsExtensions{}
  % możemy nie podawać ścieżki (dla plików w aktualnym katalogu) lub podawać ścieżkę ⤸
  relatywną
  \\ Rys. 1: podpis rysunku
}
\hspace{\stretch{1}}
 \parbox[c]{0.4\textwidth}{\centering
  % możemy też zrobić to samo nakładając na niego dodatkowy tekst
  \begin{overpic}[scale=0.44]{/usr/share/texlive/texmf-dist/tex/latex/mwe/example-image}
   \put(0,0) {\raisebox{3pt}{\colorbox[rgb]{0,.5,0}{\bf Fot. 1}}}
  \end{overpic}
  \\ Fot. 1: podpis rysunku
  \\ dwulinijkowy ;-)
```

```
}
\hspace{\stretch{2}}


\DefineShortVerb{\#} % zamiast \verb# używamy skróconej notacji ustawianej przy pomocy ⤸
komendy \DefineShortVerb z pakietu fancyvrb

Powyższe rozmieszczenie obrazków uzyskane zostało z zastosowaniem dwóch #\parbox# ⤸
pomiędzy trzema #\hspace{\stretch{x}}#
zapewnia to umieszczenie dwóch obrazków (wraz z ich podpisami) obok siebie.

#\stretch{x}# podaje wartość rozciągliwego odstępu poziomego w proporcjach ⤸
odpowiadających podanym argumentom
(#\hspace{\stretch{x}}# jest podobne do kilkukrotnego zastosowania #\hfill# i #\hfil#).
\textbf{Uwaga:} Działają one w ramach akapitu a nie wiersza w akapicie, czyli użycie ⤸
łamania linii (#\\#) może mieć na nie zły wpływ.

#\parbox# wstawia pionowe pudełko (elementy w nim są układane jeden nad drugim) o zadanej ⤸
szerokości. Podobnie działa środowisko "minipage".
Jeżeli potrzebujemy pudełka o automatycznie dobranej szerokości możemy użyć #\pbox# i ⤸
środowiska "varwidth".

\begin{Example}
 \newsavebox{\fmbox}
 \begin{lrbox}{\fmbox}
  \begin{varwidth}{0.4\textwidth}
   \hspace{\stretch{2}} aaa \hspace{\stretch{2}} \par
   bbb \par
   bbb ccc ddd
  \end{varwidth}
 \end{lrbox}
 \fbox{\usebox{\fmbox}}
\end{Example}


\subsection{Zapamiętywanie pudełek i manipulowanie nimi}

W przykładzie zostało użyte środowisko lrbox, które jest jedną z możliwości ⤸
zapamiętania pudełka do użycia w przyszłości (poprzez komendę #\usebox{PUDELKO}#).
W tym wypadku zostało to robione aby użyć #\fbox# na zawrtości "varwidth", celem objęcia ⤸
go ramką.
Innym zastosowaniem zapamiętanych pudełek jest ich wielokrotne użycie lub zamiana ich ⤸
rozmiarów post factum.
Nie wpływa ona w żaden sposób na zawartość pudełka, ale ma bardzo duży wpływ na ⤸
rozpychanie lub nie przez pudełko sąsaidujących elementów.

\begin{Example}
 \begin{lrbox}{\fmbox} \fbox{\hbox{QWERTY}} \end{lrbox}
 aaa \usebox{\fmbox} bbb \par
 \wd\fmbox=0pt aaa \usebox{\fmbox} bbb \par
 \wd\fmbox=70pt aaa \usebox{\fmbox} bbb
\end{Example}

Podobnie możemy postępować z wysokością pudełek (#\ht\PUDELKO# i #\dp\PUDELKO#). ⤸
Pakiety takie jak \href{http://www.ctan.org/pkg/calc}{calc}, \href{http://www.ctan.⤸
org/pkg/etoolbox}{etoolbox} ułatwiają bardziej zaawansowane operacje na wymiarach ⤸
elementów.


\subsection{Przycinanie i poszerzanie}
```

```
Podobne operacje przycinania pudełek możemy wykonać także dzięki trimbox z \href{https:
//www.ctan.org/tex-archive/macros/latex/contrib/adjustbox}{adjustbox}.
\begin{Example}
BBBB\trimbox{0.25cm 0pt 0.5cm 0pt}{
  \colorbox[rgb]{0,1,1}{AAAA}}CCCC \\
BBBB\clipbox{0.25cm 0pt 0.5cm 0pt}{
  \colorbox[rgb]{0,1,1}{AAAA}}CCCC \\
BBBB\marginbox{0.25cm 0pt 0.5cm 0pt}{
  \colorbox[rgb]{0,1,1}{AAAA}}CCCC
\end{Example}

Jak widać w powyższym przykładzie dostępny jest też clipbox powodujący obcięcie 
zawartości pudełka oraz marginbox o działaniu odwrotnym - powiększa pudełko dodając do 
niego marginesy.


\subsection{Obroty, odbicia i skalowanie}

Pakiet \href{https://www.ctan.org/pkg/graphics}{graphics} oprócz możliwości wstawiania 
obrazków udostępnia kilka przydatnych pudełek:
\vspace{-0.66\parskip}\begin{easylist}\initPointsList
& #\rotatebox# (obraca podaną zawartość względem wskazanego punktu o wskazany kąt),
& #\scalebox# (skaluje n-krotnie w szerokości i m-krotnie w wysokości)
& #\resizebox# (skaluje do zadanego rozmiaru w szerokości i/lub wysokości, użycie zamaist 
któregoś z rozmiarów ! pozwala na skalowanie proporcjonalne)
\end{easylist}\vspace{-0.66\parskip}\noindent
Większość tych operacji może być także wykonana bezpośrednio na wstawianej grafice 
poprzez opcje #\includegraphics#.

tekst \rotatebox{45}{obrócony napis} tekst \rotatebox[x=1cm,y=2mm]{-45}{\pbox{4cm}{obrócony 
napis\\ w drugą stronę\\i względem\\innego punktu}} tekst \rotatebox{180}{oraz do góry 
norami}

tekst \scalebox{3}[1]{rozciągnięty} tekst \scalebox{-1}[1]{odbity} tekst \scalebox{1}[-1]{
inaczej odbity}


\subsection{Kolorowe pudełka}
W jednym z powyższych przykładów jako obiektu przycinanego użyto pudełka #\colorbox# z 
pakietu \href{https://www.ctan.org/pkg/color}{color} powodującego wyświetlenie jego 
zawartości na tle o zadanym kolorze (kolor można określać m.in. przez nazwę, czy też 
składowe rgb). Pakiet ten pozwala także na pokolorowanie tekstu (co było już pokazywane), 
tła całej strony (co za chwilę zostanie zademonstrowane) oraz m.in. wstawianie pudełek z 
kolorową ramką \fcolorbox{red}{green}{TEKST} poprzez #\fcolorbox{kolor ramki}{kolor tła}{
ZAWARTOŚĆ}#.


\subsection{Jeszcze więcej pudełek}

Bardzo wszechstronnym pudełkiem jest #\adjustbox# z pakietu \href{https://www.ctan.org/tex-
archive/macros/latex/contrib/adjustbox}{adjustbox} który łączy przedstawione wyżej 
możliwości takie jak skalowanie, obcinanie itp w jednym pudełku. Pozwala także na 
określanie grubości obramowania

tekst tekst \adjustbox{cfbox=red 3pt 0pt 5mm}{\pbox{10cm}{grube czerwone obramowanie\\ bez 
wewnętrznego marginesu,\\ ale za to z dużym zewnętrznym}} tekst \adjustbox{raise=2ex}{
uniesiony} tekst

wyrównania w pionie ...\adjustbox{valign=T}{\pbox{1cm}{T\\b\\c}}.\adjustbox{valign=M}{\pbox{
```

```
1cm}{M\\b\\c}}.\adjustbox{valign=B}{\pbox{1cm}{B\\b\\c}}...
...\adjustbox{valign=t}{\pbox{1cm}{t\\b\\c}}.\adjustbox{valign=m}{\pbox{1cm}{m\\b\\c}}.
\adjustbox{valign=b}{\pbox{1cm}{b\\b\\c}}...
pionie i poziomie ...\adjustbox{stack=cm, bgcolor={rgb}{0 1 0}}{stack\\b\\c}... wraz z
kolorkiem tła

Pakiet ten pozwala także na tworzenie pudełek będących uniwersalnym odstępem pionowym
i/lub poziomym poprzez:\\
#\phantombox{szerokość}{wysokość - od linii w górę}{głębokość - od linii w dół}#.
Jako tła dla pudełka możemy używać pliku graficznego:

\adjustbox{cfbox=black 1pt 4mm, bgimage=/usr/share/texlive/texmf-dist/tex/latex/mwe/example-
image-16x10.png, minipage=10cm}{\color{white}\lipsum[2]}

Możemy także wykorzystać adjustbox do zdefiniowania własnych typów pudełek - np.
cieniowanych czy z podwójnym obramowaniem:

\UndefineShortVerb{\#} % gdy już nie potrzebujemy skróconej notacji \verb wyłączamy ją,
aby uniknąć potencjalnych problemów z innymi instrukcjami

\begin{ExampleVertical}
% pudełko z podwójną niebiesko-czerwoną ramką
\def\dframeBox#1{\adjustbox{cfbox=blue 0.5pt 1pt}{\adjustbox{cfbox=red 0.3pt}{#1}}}
% pudełko z zielonym cieniem
\def\shadowBox#1{%
  \newlength{\boxWidth}%
  \hbox{%
    \hspace{4pt}%
    \adjustbox{cfbox=green, bgcolor=green, raise=-4pt, gstore width=\boxWidth}
      {\color{green}#1}%
    \setlength{\boxWidth}{\boxWidth + 4pt}\hspace{-\boxWidth}%
    \adjustbox{cfbox=red 0.3pt, bgcolor=white}{#1}%
    \hspace{4pt}%
  }%
  \undef{\boxWidth}%
}
xx \dframeBox{AAAA} xx \shadowBox{AAAA} xx
\end{ExampleVertical}


\subsection{Linie poziome}
Dosyć gruba, wyśrodkowana linia o szerokości 50\% tekstu:\par
\hfil\rule{0.5\linewidth}{1pt}\hfill

Jeszcze grubsza, przerywana w układzie 4mm linii, 5mm przerwy, 2 mm linii, 1mm przerwy o
szerokości całego tekstu:\par
\hdashrule{\linewidth}{2pt}{4mm 5mm 2mm 1mm}


\subsection{Włączanie wielostronicowych plików PDF}

Pakiet \href{https://www.ctan.org/pkg/pdfpages}{pdfpages} umożliwia włączanie plików PDF
lub ich części do generowanego dokumentu. Pozwala na wybór stron z włączanego pliku,
umieszczanie kilku stron na jednej stronie, dodawanie stron z włączanych plików do spisu
treści, automatyczne skalowanie dodawanych stron lub dostosowywanie rozmiarów stron do
dodawanych stron i wiele więcej.

W office.cls zdeklarowane zostały także makra ułatwiające dodawania załączników przy
pomocy pdfpages do dokumentu w tym celu wystarczy dodać:
\begin{minted}{tex}
```

```
% wstawienie spisu załączników
\spisZalacznikow

% deklaracje kolejnych załączników
\zalacznik{nazwa}[opcje]{opis}
% gdzie:
%   nazwa - którtki identyfikator załącznika a jednocześnie nazwa pliku bez rozszerzenia
%   opcje - opcje \includepdf np. fitpaper=false lub angle=90
%   opis  - opis załącznika nadrukowywany na nim i występujący w spisie załączników
\end{minted}

\includepdf[fitpaper=true]{/usr/share/texlive/texmf-dist/tex/latex/mwe/example-image-a5-⤸
landscape.pdf}


% wymuszamy podział strony i modyfikujemy ustawienia geometrii - zmieniamy orientację (⤸
moglibyśmy także np. wielkość) strony
\forceNewPageGeometry{landscape,a4paper}
%
\subsection{Geometria strony}
%
% na tej stronie nie umieszczamy nagłówka i stopki
\thispagestyle{empty}
%
% dodajemy na tej stronie zakładki "fancytabs"
% więcej info: http://mirrors.ctan.org/macros/latex/contrib/fancytabs/fancytabs.pdf
\fancytab{Tab A}{1} \fancytab{Tab B}{3}
%
% kolor tła całej strony
\pagecolor[rgb]{0.9,1,0.9}

Ta strona celowo jest zielonkawa dzięki \verb#\pagecolor#.
Wysokości: textheight=\the\textheight{} vsize=\the\vsize{}
Szerokości: textwidth=\the\textwidth{} headwidth=\the\headwidth{} hsize=\the\hsize{} ⤸
linewidth=\the\linewidth{}

\namedLabel{test1}{etykieta na obróconej stronie}

% środowisko obrazka opływanego tekstem,
%  jego położenie (r - do prawej, l - do lewej, ...) i rozmiar
%  jest też kilka opcjonalnych argumentów
\begin{wrapfigure}{r}{8cm}
  % wyśrodkowanie zawartości pola obrazka
  \begin{center}
    % okienko skalujące:
    %  pierwszy argument szerokość, drugi wysokość,
    %  jeden z nich może być zastąpiony ! - zachowanie proporcji obrazka
    %  w taki sposób możemy skalować także inne obiekty np. tekst
    % alternatywnie możemy też użyć \scalebox{0.3}{ ... } określającego skalowanie ⤸
    względne
    \resizebox{4cm}{!}{
      % wstawienie obrazka
      \includegraphics{/usr/share/texlive/texmf-dist/tex/latex/mwe/example-grid-100x100bp.png}
    }
    % opis obrazka
    \caption[Opcjonalny skrócony opis do spisu obrazków]{Podpis do obrazka}
    % etykieta
    \label{etykietka_obrazka}
  \end{center}
\end{wrapfigure}
```

```
\begin{spacing}{1.7}
\textbf{To są akapity z zwiększonymi odstępami międzyliniowymi.}
\lipsum[1]
\namedLabel{test2}{inna etykieta na obróconej stronie}
\lipsum[2]
\end{spacing}

% wymuszamy podział strony i modyfikujemy ustawienia geometrii - zmieniamy orientację i
przechodzimy na tryb dwustronny
\forceNewPageGeometry{portrait,a4paper,twoside=true}
% wyłączamy kolor tła strony
\nopagecolor


\section{Tabele}
\begin{ExampleVertical}
 \begin{tabular}{ p{2cm} | l r c r@{.}l }
  % spacje w kodzie tabelki tylko dla czytelnego pokazania kolumn
  AAAA &                  BBB BBBB & CC      & ddd dd        & 0&12356      \\ \hline
  aaaa aa bbb ddd eee & bbb      & \multicolumn{2}{c}{DDD} & 112&57       \\
  QQQQ &                  WWW W    & EEE     & RRRR          & 98765&12356  \\
\end{tabular}
\end{ExampleVertical}

Powyżej pokazana jest prosta \LaTeX'owa tabelka. Wiele pakietów rozszerza możliwości
konfiguracyjne tabel np. pozwalają używać komórek wielowierszowych, pozwalają na
formatowanie liczb (w szczególności pozwala na prawdziwe, a nie uzyskane pokazaną wyżej
sztuczką, wyrównanie liczb względem separatora dziesiętnego). Poniższa tabelka korzysta
z pakietów:
\href{http://mirrors.ctan.org/macros/latex/required/tools/array.pdf}{array},
\href{http://mirrors.ctan.org/macros/latex/contrib/colortbl/colortbl.pdf}{colortbl},
\href{http://mirrors.ctan.org/macros/latex/required/tools/hhline.pdf}{hhline},
\href{http://mirrors.ctan.org/macros/latex/contrib/diagbox/diagbox.pdf}{diagbox},
\href{http://mirrors.ctan.org/macros/latex/contrib/siunitx/siunitx.pdf}{siunitx} (dla
wyrównania względem separatora dziesiętnego)
oraz multirow.

\begin{ExampleVertical}
 \begin{tabular}{
    || >{\raggedleft}p{3cm} || r
    || >{\color{yellow}\columncolor[gray]{.2}[.5\tabcolsep]}c
     | S !{\color{blue}\vrule width 3pt}
 }
  \hhline{|t:==:t:==|}
    \diagbox[width=3.4cm]{AA}{BB}{CC} &
    BBB &
    \multicolumn{1}{c}{ddd dd} & % komórka bez obramowania z prawej
    0.12356 \tabularnewline % działa jak \\ ale bywa bezpieczniejszy dla formatowania
  \hhline{|:=||=>{\arrayrulecolor{red}}=>{\arrayrulecolor{black}}=|}
    \textcolor{red}{aaaa \textbf{aa bbb ddd} eee} &
    \multirow{2}{2.5cm}[-0.75cm]{bbb aaaa aa bbb ddd} &
    ccc ccc &
    112.57 \tabularnewline
  \hhline{#=#~||~|->{\color{blue}\vrule width 3pt}}
    % sztuczka z >{...} zapewnia zgodną z definicją w nagłówku linię po prawej
    QQQQ \newline od nowej linii &
    & % ta komórka powinna być putsta bo wyżej użyliśmy \multirow
    \diagbox[dir=SE]{AA}{CC} &
    \cellcolor[rgb]{0,1,1} 98765.12356 \tabularnewline
```

```
  \hhline{% !{...} jest rozszerzeniem \hhline zdefiniowanym w office.cls
     !{\leaders\hrule height 1pt\hfil}%    pozwala na tworzenie dowolnych
     !{\leaders\hbox{\tiny$\star$}\hfil}%  poziomych linii rozdzielających
     !{\hfil...\hfil...\hfil}%
     !{\leaders\hbox{\hdashrule{1.5mm}{1pt}{0.5mm 0.5mm 0.5mm 0mm}}\hfil}%
  }
 \end{tabular}
\end{ExampleVertical}


Jeżeli chcemy korzystać w ramach tabu z komórek rozciągniętych równocześnie na kilka ⤸
kolumn i wierszy możemy deklarować je w sposób następujący:
\begin{InlineVerbatim*}
\multicolumn{2}{c|}{
 \multirow{3}{2\tabucolX}{\centering
  zawartość komórki
}
}
% nawet przy zwykłym multirow w kolumnach deklarowanych jako X
% należy jako drugi argument podawać \tabucolX ... tutaj mamy
% 2\tabucolX bo zajmujemy miejsce dwóch kolumnt typu X z powodu \multicolumn
\end{InlineVerbatim*}


\newpage

\subsection{Wyrównanie w pionie i poziomie}

Poniższa tabelka bazuje na zdefiniowanych w office.cls typach kolumn i komórek (⤸
wielowierszowych, wielokolumnowych, niezależnie formatowanych) - umożliwiają one poprawne ⤸
wyrównywanie wysokości elementów w tabeli. Użyty longtable pozwala na dzielenie tej ⤸
tabelki pomiędzy strony.

\begin{ExampleVertical}
\begin{longtable}{
    b | C{2cm}{}{t} | C{2cm}{1.5cm}{m} | C{2cm}{}{b} | L{2cm}{}{m} |
    R{2cm}{}{t} | J{2cm}{}{t} I{\color{red}}{1pt}{0.5mm 0.5mm 0.5mm 0mm} e
  }
    top &
    middle &
    bottom &
    middle left &
    top right &
    top justify
  \\ \hline
    A \lb xxx \lb X &
    B &
    C &
    \setmultirow{3}{
      ala ma kota \lb kot ma ale la la \lb kot ma ale la la \lb kot } &
    q &
    q
  \\ \hhline{---~--}
    D \lb d &
    E \lb xxx xx \lb X \lb X &
    F \lb f &
    &
    q &
    q
  \\ \hhline{---~--}
```

```
    G &
    H &
    I \lb xxx \lb Xj &
    &
    q &
    q
  \\ \hline
    G &
    \mymulticolumn{2}{4cm}{3cm}{c}{t}{|}
      {top, center: I I \lb ala ma kota, kot ma ale ala ma kota, kot} &
    xx &
    q &
    q
  \\ \hline
    G &
    \multicolumn{2}{Z{4cm}{3cm}{l}{2}{t}{1}|}
      {top, left: I I \lb xxxxxxx \lb a a Xj} &
    LL LL\lb xxx\lb X\lb X &
    RR RR\lb xxx\lb X\lb X &
    BB BB\lb xxx\lb X\lb X
  \\ \hline
    G &
    \tableFormatedCell{3.5cm}[3cm]{r}{b}I I \lb xxxxxxx \lb a a Xj &
    y &
    ala ma kota, kot ma ale &
    ala ma kota, kot ma ale &
    ala ma kota, kot ma ale
  \\ \hline
\end{longtable}
\end{ExampleVertical}
\vspace{-1.5cm}


\subsection{Jeszcze więcej tabelek}

Istnieje także wiele innych pakietów bardziej gruntownie modyfikujących składanie tabel (
jak choćby mdwtab, longtable czy tabularx). Najbardziej wszechstronnym i elastycznym wydaje
się \href{http://mirrors.ctan.org/macros/latex/contrib/tabu/tabu.pdf}{tabu}. Powyższe
tabelki są z nim kompatybilne (to znaczy pokazane efekty obramowań, wyrównań,
rozciągniętych komórek itd możemy uzyskać w ten sam sposób).

\begin{ExampleVertical}
 \taburowcolors 2{gray!15 .. gray!50}
 \begin{tabu} {|[1pt blue]|[1pt green] m{2cm} || r || c | S }
  \tabucline[3pt on 1.5pt blue off 2pt red]{1-2}
  \tabucline[2pt white]{1-2}
  \tabucline[3pt on 1.5pt blue off 2pt red]{1-2}
    \diagbox[width=2.4cm]{AA}{BB}{CC} &
    BBB &
    ddd dd &
    0.12356 \\
  \hhline{|:=||=>{\arrayrulecolor{red}}=>{\arrayrulecolor{black}}=}
     aaaa aa bbb ddd eee &
     bbb aaaa aa  &
    ccc ccc &
    112.57 \\
  \tabucline[2pt red]{1-1}
  \tabucline[\hbox{\textcolor{blue}{x}}]{1-1}
  \tabucline[2pt red]{1-1}
    QQQQ &
```

```
    bbb ddd eee &
    \diagbox[dir=SE]{AA}{CC} &
    98765.12356 \\
\end{tabu}
\end{ExampleVertical}

Dzięki pakietowi \href{http://www.ctan.org/pkg/spreadtab}{spreadtab} można nawet używać ↩
prostych arkuszy kalkulacyjnych.
\begin{Example}
 \begin{spreadtab}{{tabular}{ccc}}
  @A/1 & @B                 & @C       \\
  2    & 11                 & 23       \\
  3    & @ b2 + c2 =   & b2 + c2  \\
  4    & 54                 & 10       \\
  5    & :={c4} * c3 =      & c4 * c3
\end{spreadtab}
\end{Example}

Tabele możemy umieszczać także poza głównym tekstem (analogicznie jak obrazki) poprzez ↩
wstawienie ich w środowisko "table". Dzięki użyciu \href{http://mirrors.ctan.↩
org/macros/latex/contrib/threeparttable/threeparttable.pdf}{threeparttable} możliwe jest ↩
umieszczanie w takich tabelech niezależnych przypisów.


\section{Podziały}
\DefineShortVerb{\#}
Oprócz wymuszania podziału linii (#\\# lub #\newline#), paragrafu (#\par#), strony (↩
#\newpage# lub #\clearpage# możliwe jest także zniechęcanie/zachęcanie do podziału linii ↩
lub strony w danym miejscu poprzez: #\linebreak[x]#, #\nolinebreak[x]#, #\pagebreak[x]# oraz ↩
#\nopagebreak[x]#, gdzie x jest liczbą z zakresu od 0 do 4 i oznacza stopień ↩
zachęcania/zniechęcania (4 jest wartością domyślną i oznacza wymuszenie). #\linebreak[4]↩
# w odróżnieniu od #\\# powoduje rozciągnięcie ostatniego wiersza.

Możemy także zapobiegać łamaniu linii na spacji poprzez użycie zamiast spacji #~#. oraz ↩
zapobiegać łamaniu całych wyrażeń umieszczając je w #\hbox#.
\UndefineShortVerb{\#}



\section{Odnośniki}
\DefineShortVerb{\#}
Możemy tworzyć odnośniki wewnętrzne z wykożystaniem #\label{ID}# (lub zdefiniowanego w ↩
office.cls  #\namedLabel{ID}{nazwa}#) w miejscu na które wskazujemy i #\ref{ID}# lub ↩
#\hyperref[ID]{tekst}# w miejscu gdzie chemy umieścić odnośnik.
Możliwe jest także tworzenie odnośników zewnętrznych poprzez #\url{URL}# (\url{http:↩
//www.opcode.eu.org/}) lub #\href{URL}{tekst}# (zademonstrowane kilkakrotnie w tym dokumencie)↩
.\\
Link do: \ref{test1} oraz \hyperref[test2]{inny link do tej samej strony w hyperref}. Warto ↩
także wspomnieć o przypisach\footnote{to jest przypis}, które możemy dodawać w tekście. ↩
Są one umieszczane na dole strony.

Z pomocą pakietu \urllink[border=1, color=1 1 0, dash=3 2]{http://www.ctan.org/tex-↩
archive/macros/generic/navigator}{navigator} możliwe jest także tworzenie bardziej surowych ↩
linków PDFowych poprzez #\anchor{ID}#, odwoływać się do nich można poprzez: #\jumplink{↩
ID}{tekst linku}#. W odróżnieniu od #\label{ID}# prowadzą one do konkretnego miejsca na ↩
stronie - \jumplink[border=1]{test1}{link na górę obróconej strony} oraz \jumplink[↩
pre=\bgroup\itshape,post=\egroup,linkcolor=1 0.5 0]{test2}{link na dół obróconej strony}. ↩
\bs anchor pozwala także wskazanie ofsetu od miejsca umieszczenia identyfikatora oraz ↩
wymaganego powiększenia dokumentu.
```

```
Możliwe jest także dodawanie pozycji do zakładek poprzez #\outline{x}[outlineID]{tekst}# (⤸
definiuje etykietę ID w miejscu wstawienia) lub #\outline[anchor=ID]{x}{tekst}# (zakładka ⤸
wskazuje na etykietę ID), gdzie x określa poziom w menu zakładek. Zakładki te zostaną ⤸
umieszczone po zakładkach związanych z automatycznym spisem sekcji.\\
\outline[anchor=NON,open=true]{1}{zakładki navigator}\outline[anchor=test2]{2}{dół ⤸
obróconej strony}
Możliwe jest osadzanie plików w dokumencie pdf poprzez:
\begin{InlineVerbatim}
 \embeddedfile
   [opis]{identyfikator}
   [nazwa_widoczna.w.pliku.pdf]{nazwa_pliku-do.osadzenia}
\end{InlineVerbatim}
 \embeddedfile[definicja klasy dokumentu]{office.cls}{office.cls}
Ponaddto pakiet ten umożliwia również wstawianie odnośników zewnętrznych poprzez ⤸
#\urllink#.
\UndefineShortVerb{\#}


\newpage

\section{Spisy treści}
\DefineShortVerb{\#}
Spis treści umieszczamy poprzez wstawienie #\tableofcontents# aby nadać mu nazwę "Spis ⤸
treści" korzystamy wcześniej z #\def\contentsname{Spis treści}#. Pozycję do spisu ⤸
wprowadzają automatycznie polecenia takie jak: chapter, section, subsection, ..., możemy ⤸
też wstawić ją ręcznie:
\begin{InlineVerbatim}
  \addcontentsline{toc}
    {nazwa_poziomu_np_section}
    {tekst który ma zostać podany w spisie}
\end{InlineVerbatim}
Aby decydować które poziomy są pokazywane w spisie treści można użyć #\setcounter{⤸
tocdepth}{x}#, gdzie x określa maksymalny numer poziomu pokazywanego w spisie.
Podobnie możemy też dodawać informacje dodatkowe np. dodatkowy odstęp:\linebreak[3]
 #\addtocontents{toc}{\protect\vspace{1ex}}#\linebreak[3]
Jeżeli chcielibyśmy mieć kropki (....) w spisie treści między tytułem sekcji a numerem ⤸
strony możemy je uzyskać przy pomocy:
\begin{InlineVerbatim}
 \makeatletter
  \renewcommand*\l@section{\@dottedtocline{1}{0cm}{0cm}}
 \makeatother
\end{InlineVerbatim}
Jeżeli chcielibyśmy pozbyć się numeracji w spisie treści możemy użyć:
\begin{InlineVerbatim*}
 \makeatletter
 \renewcommand*\mw@markandtoc{
   \addcontentsline{toc}{\mw@HeadingType}{\HeadingTOCText}
 }
 \makeatother
\end{InlineVerbatim*}
\UndefineShortVerb{\#}



\section{Informacje dodatkowe}

\subsection{Komentarze}
Oprócz klasycznych komentarzy (od \verb@%@ do końca linii) możemy też użyć komentarza ⤸
blokowego:
\begin{InlineVerbatim}
```

```
  \begin{comment}
    komentarz wieloliniowy
    ALA MA KOTA
  \end{comment}
\end{InlineVerbatim}
Także tekst za \verb@\end{document}\verb@ jest ignorowany będąc swego rodzaju komentarzem.


\subsection{Matematyka}
Przedstawie tu tylko wybrane elementy związane z składem wyrażeń matematycznych - po ↩
pełwn opis (w tym listę różnych symboli) odsyłam do rozdziału trzeciego \href{ftp://ftp.↩
gust.org.pl/TeX/info/lshort/polish/lshort2e.pdf}{Nie za krótkie wprowadzenia do systemu ↩
LaTeX 2e} oraz innych źródeł.\\
dwumiany bez nawiasów / z nawiasami  - ${a \atop b}$ / ${a \choose b}$\\
ułamiki - ${a \over b}$ lub $\frac{a}{b}$\\
ułamki ukośne (zdefiniowane w office.cls) - $\minifrac{a}{b}$\\
kreska "w punkcie" np. do pochodnej (zdefiniowane w office.cls) - $\inpoint{2x}$\\
linia nad / pod jakimś wyrażeniem - $\overline{a+b}$ / $\underline{a+b}$\\
napis nad strzałką $\stackrel{a}{\rightarrow}$\\
klamra pod / nad wyrażeniem (z podpisem pod / nad klamrą) - $\underbrace{2+3}_{5}$ / ↩
$\overbrace{2+3}^{5}$

Interesujący jest też pakiet \href{http://mirrors.ctan.↩
org/macros/latex/contrib/siunitx/siunitx.pdf}{siunitx} pozwalający na formatowanie liczb i ↩
jednostek
\begin{Example}
\sisetup{group-minimum-digits=4,negative-color=red}
\num[group-separator={\,}]{1245.34}\\
\num[output-decimal-marker={~}]{1245.34}\\
\num{-12.8} \num[negative-color=]{-12.8}\\
\SI[mode=text]{1.23}{J.mol^{-1}.K^{-1}}
\end{Example}


\subsection{Bibliografia i bibtex}
Odnośniki do bibliografii wstawiamy poprzez \verb#\cite{ID_pozycji_bibliograficznej}#. Samą ↩
bibliografię możemy utworzyć:
\begin{InlineVerbatim}
 \begin{thebibliography}{99}
  \addcontentsline{toc}{chapter}{Bibliografia}
  \bibitem[identyfikatorWidoczny]{ID_pozycji_bibliograficznej}%
          Autor, \textit{Tytuł}, Miejsce publikacji i rok.
\end{thebibliography}
\end{InlineVerbatim}
Przy większych pracach warto skorzystać z systemu do tworzenia bibliografii bibtex, ↩
odwołania do pozycji tworzy się w identyczny sposób, a sam spis umieszcza się poprzez:
\begin{InlineVerbatim}
 \cleardoublepage\phantomsection%
   \addcontentsline{toc}{chapter}{Bibliografia}
 \bibliographystyle{plunsrt}\nocite{*}%
   \bibliography{nazwa_pliku_z_bibliografia}
\end{InlineVerbatim}
Budowanie latex'a z zastosowaniem bibtexa wymaga wydania dodatkowo komendy bibtex ↩
nazwa\_bez\_rozszerzenia przed budowaniem właściwego pliku latex'a. Sam plik ↩
bibliograficzny składa się z rekordów postaci (więcej na ten temat w Bibliografia w ↩
LaTeXu - program bibtex):
\begin{InlineVerbatim*}
@misc{ID_pozycji_bibliograficznej,
author        = "",
title         = "",
```

```
journal       = "",
year          = "",
volume        = "",
school        = "",
url           = "",
institution   = "",
howpublished  = "",
type          = ""
}
\end{InlineVerbatim*}


\subsection{Liczniki}
\DefineShortVerb{\#}
Liczniki przydają się do automatycznej numeracji rozdziałów, paragrafów itp. Liczniki ⤸
tworzymy poleceniem #\newcounter{nazwa_licznika}#, wartość nadajemy mu poprzez #\setcounter{⤸
nazwa_licznika}{wartosc}#, zwiększać możemy ją o jeden poprzez instrukcję #\stepcounter{⤸
nazwa_licznika}#, natomiast wstawiać przez instrukcje typu #\Roman{nazwa_licznika}# (duże ⤸
liczby rzymskie), #\Alph{nazwa_licznika}# (duże litery), #\alphalph{nazwa_licznika}# (małe ⤸
litery  - wersja obsługująca numerację powyżej z jako aa, ab, ac, ...), #\arabic{⤸
nazwa_licznika}# (liczby arabskie), ... .
Poniższy przykład ilustruje sposób umieszczania automatycznych odwołań do zadanych ⤸
fragmentów pliku (np. paragrafów jakiegoś regulaminu).
\UndefineShortVerb{\#}
\begin{InlineVerbatim*}
% wlaczam plik z definicjami liczników
\IfFileExists{\jobname.cou}{\input{\jobname.cou}}{}

% otwieram do zapisu plik z baza odnośników i zapisuje nagłówek
\newwrite\licznfile
\openout\licznfile=\jobname.cou
\write\licznfile{\string\def\string\liczniki{}}

% ustawiamy zamknięcie plik z licznikami w /end{document}
\AtEndDocument{\closeout\licznfile}

% instrukcja zapamiętuje licznik w pliku (jako licznik o zadanej nazwie)
% \zapamietajlicznik{nazwa_licznika_do_zapamietania}{nazwa_nowego_licznika}
% przy czym dla środowiska enumerate są to dla kolejnych poziomów:
%  enumi enumii enumiii enumiv ...
\newcommand{\zapamietajlicznik}[2] {
  \immediate\write\licznfile{
    \string\newcounter{#2}\string\setcounter{#2}{\arabic{#1}}
  } % \string - zabezpiecza backslesh (\)
}

% nie tworze liczników przy pierwszym obiegu - zamiast nich będę wpisywał XXX
% dopiero po wczytaniu ich z pliku wstawię odpowiednie numerki
\newcommand{\alphf}[1]{\ifx \liczniki \undefined XXX  \else\alph{#1}\fi}
\newcommand{\arabicf}[1]{\ifx \liczniki \undefined XXX  \else\arabic{#1}\fi}

% W miejscu do którego chcemy się odwołać
% umieszczamy instrukcję zapamiętania licznika:
%  \zapamietajlicznik
%    {nazwa_zapamietywanego_licznika}
%    {nazwa_pod_ktora_chemy_go_zapamietac}.
%
% W miejscu w którym chcemy wstawić odwołanie umieszczamy np:
%  \arabicf{nazwa_pod_ktora_zapamietalismy_licznik}
\end{InlineVerbatim*}
```

```
\subsection{Zaawansowana grafika z użyciem Ti\textit{k}Z / PGF}

Jest to system do tworzenia zaawansowanych rysunków z użyciem \LaTeX'a. Oprócz głównej ⤸
biblioteki - \href{http://www.ctan.org/pkg/pgf}{Ti\textit{k}Z / PGF} istnieje wiele ⤸
specjalistycznych \href{http://www.ctan.org/tex-archive/graphics/pgf/contrib}{pakietów z ⤸
niej korzystających} - np:
\vspace{-0.66\parskip}\begin{easylist}\initPointsList
& \href{http://www.ctan.org/tex-archive/graphics/pgf/contrib/pgfplots}{pgfplots} - umożliwia ⤸
generowanie wykresów 2D (w tym histogramów) i 3D
& \href{http://www.ctan.org/tex-archive/graphics/pgf/contrib/pgf-pie}{pgf-pie} - umozliwia ⤸
generowanie wykresów kołowych
\end{easylist}
Poniżej tylko kilka prostych przykładów zaczerpniętych z dokumentacji pakietów ⤸
używających pgf (oficjalna dokumentacja samego Ti\textit{k}Z / PGF liczy ponad 1000~stron).

\begin{Example}
  \begin{tikzpicture}
    \node [minimum size=3.13cm,
    decorate, decoration=zigzag,
    fill=blue!20,draw,thick,circle] {Hello!};
  \end{tikzpicture}
\end{Example}

\begin{Example}
  \begin{tikzpicture}
    \node [circle,draw] (A)
    {A};
    \node [circle,draw] (B) [right=of A] {B};
    \draw [draw = blue, thick]
    (A) edge [bend left=45] (B)
    (B) edge [in=-110, out=-70,looseness=8] (B);
  \end{tikzpicture}
\end{Example}

\begin{Example}
  \begin{tikzpicture}
    \usetikzlibrary{graphs}
    \graph {
      4 -- {
      3 -- 0 -- 1,
      10 -- {
      8 -- {
      6 -- {5,7},
      9
      } } }
    };
  \end{tikzpicture}
\end{Example}

\begin{center}
\begin{tikzpicture}
\begin{loglogaxis}
 \addplot coordinates {
  (1,1)
  (16,16)
  (32,64)
};
\end{loglogaxis}
```

```
\end{tikzpicture}
\end{center}


\clearpage\section{Zaawansowane funkcje PDFów itp}

\subsection{Interaktywne formularze}
Pakiet hyperref zapewnia wsparcie dla tworzenia interaktywnych formularzy w PDFach. Dodatkowe ↩
funkcjonalności związane z ich obsługą dostarcza \href{http://ctan.org/pkg/acrotex}{↩
Acro\TeX}. Poniżej pokazany jest przykład formularza stworzonego z użyciem hyperref.
\begin{Example}
 \renewcommand{\DefaultHeightofText}{10pt}
 \renewcommand{\DefaultHeightofChoiceMenu}{10pt}
 \begin{Form}
  \TextField[default=kota]{Ala ma: }\\
  \TextField[multiline=true,height=20pt]{Pies ma: }\\
  \ChoiceMenu[combo]{Wybierz: }%
    {a1, b5, j8, wdfed}\\
  \CheckBox{Akceptacja}
 \end{Form}
\end{Example}

\subsection{Notatki PDF}
\begin{pdfsidelinecomment}[color=red,author={Inny Autor},linewidth=4mm,linesep=1cm]{ PDF note ↩
}
Pakiet \href{http://www.ctan.org/pkg/pdfcomment}{pdfcomment} jest chyba najbardziej ↩
rozbudowanym z pakietów umożliwiających dodawanie do plików pdf różnego rodzaju notatek.↩
Oprócz niego warto zwrócić uwagę na pakiety attachfile i marginnote, pozwalające ↩
odpowiednio na: dodanie notatki zawierającej załącznik plikowy i umieszczenie takiej ↩
notatki na marginesie (poprzez umieszczenie komendy ją wstawiającej w argumencie ↩
\verb#\marginnote{}#.

\begin{Example}
\attachfile[%
 author={Autor Notatki}, subject={Tytul notatki},%
 description={Treść notatki}, icon={Tag},%
 mimetype={text/plain}]{\jobname.tmp}
\end{Example}
\end{pdfsidelinecomment}

Z użyciem pdfcomment możemy komentować bloki tekstu (tak jak powyżej po bokach), ↩
wstawiać notatki na marginesach oraz w \pdfcomment[icon=Note,color=yellow,author={Inny Autor}↩
]{Notatka umieszczona w tekście} tekście.
\pdfmargincomment[icon=Insert,color=red,author={Jeszcze Inny Autor}]{Notatka umieszczona na ↩
marginesie}
Możliwe jest też oznaczanie i komentowanie fragmentów tekstu poprzez
\pdfmarkupcomment[color=blue,markup=Highlight,author={Inny Autor 2}]{podświetlenia}{notatka ↩
do oznaczonego fragmentu nr 1},
\pdfmarkupcomment[color=blue,markup=Underline,author={Inny Autor 2}]{podkreślenia}{notatka ↩
do oznaczonego fragmentu nr 2} i
\pdfmarkupcomment[color=blue,markup=StrikeOut,author={Inny Autor 2}]{przekreślenia}{notatka ↩
do oznaczonego fragmentu nr 3} - niestety wydaje się że działa tylko w pdflatex (a nie ↩
działa w xelatex i lualatex).
Pakiet pozwala \pdftooltip{także na dodawanie "tooltips"}{to jest właśnie dymek związany ↩
z tamtym tekstem.}

\pdffreetextcomment[subject={free text comment},height=1cm,width=3cm,voffset=7cm,hoffset=10cm,↩
opacity=0.5,color=yellow,author={Jeszcze Inny Autor}]{Notatka w prostokącie}\vspace{-1.2cm}
```

```
\subsection{Postscript}
LaTeX umożliwia także zaawansowane zabawy z włączanymi plikami postscriptowymi. Przy ⤸
użyciu pakietu "psfrag" możemy dokonywać podmiany napisów w EPS:
\begin{InlineVerbatim}
  \psfrag{tekst_do_zastapienia}{
  tekst zastępujący \small{może} zawierać np. $wzory$
  }
  \psfragscanon
  \nopagebreak\newline\centerline{
    \includegraphics[scale=0.7]{plot.eps}
  }
  \psfragscanoff
\end{InlineVerbatim}

Możliwe jest także mieszanie PS i Latex (umieszczanie komend PS wewnątrz pliku latexowego):
\begin{InlineVerbatim}
  \psset{linewidth=1mm}
  \begin{pspicture}(3,2)
    \pscurve[arrows=<->](0,1.3)(0.7,1.8)
    (3.5,0.7)(4,1.9)(0.4,0.4)
  \end{pspicture}
\end{InlineVerbatim}



\forceNewPageGeometry{lmargin=1.5cm,rmargin=1.5cm}
\appendix
\section{Kod źródłowy}

Kod źródłowy w postaci plików tekstowych jest załączony do tego dokumentu PDF w postaci ⤸
"Embedded File".
Poniżej zaprezentowany jest kolorowany kod źródłowy wygenerowany z użyciem minted+⤸
pygments i highlight.

\subsection{office.cls}\vspace{0.5\parskip}
\inputminted[fontsize=\tiny,linenos=false]{tex}{office.cls}

\subsection{office-demo.tex}\vspace{0.5\parskip}
\begin{spacing}{0.75}
 {\tiny\highlight{office-demo.tex}}
\end{spacing}

\end{document}
```

Linki

Zachęcam także do zapoznania się z dwoma moimi dokumentami przygotowanymi w LaTeX: Curriculum Vitae[1142] i wizytówka[1143].

Przydatne pakiety i makra: tablice Karnaugha[1144], grafy itp[1145] (pakiet wykorzystałem w dokumentacji do projektu podstawa silnika 3d - jego użycie można zobaczyć w plikach tex wchodzących w skład źródeł projektu), CV[1146].

Należy także wspomnieć o dodatkowych narzędziach operujących na plikach latexowych takich jak: `latex2html`, czy też służących do zamiany równań latex'owych na MathML[1147] oraz PNG **blahtexml**[1148] lub jego pythonowy odpowiednik (tylko MathML) latex2mathml[1149].

Zachęcam także do zajrzenia do The TeX Catalogue Online, CTAN Edition[1150]. Warto również skorzystać z: Latex @ Wikibooks[1151]Łagodne wprowadzenie do TEX-a[1152]Włączanie grafik...[1153], Tworzenie dokumentów PDF przy pomocy LaTeX-a[1154], Lepszy skład polsko języczny - mwcls[1155], W miare krótki i praktyczny kurs LaTeX-a w e^pi minut[1156], Diagramy w LateX[1157], Skład wyrażeń matematycznych w LateX[1158].

Teoria informatyki

Podstawy teoretyczne informatyki to w zasadzie duży dział matematyki poświęcony teorii automatów[1159], teorii informacji[1160], teorii grafów[1161], językom formalnym i gramatykom[1162], algorytmom[1163] , itp (w zasadzie informatykę teoretyczną można uznać za dział matematyki). Z bardziej specyficznych dla informatyki działów należy wspomnieć o teorii obliczeń[1164], terorii kompilacji (blisko związanej z językami formalnymi) i strukturach dancyh[1165] (blisko związanych z algorytmiką[1166]). Informatyka teoretyczna zajmuje się w dużej mierze konstrukcją algorytmów, dowodzeniem ich poprawności[1167] i analizą złożoności obliczeniowej[1168]. Zobacz także: Logika Hoare'a[1169], Niezmiennik pętli[1170], Analiza składniowa[1171], Parser LR[1172], LL parser[1173]. Często do teorii informatyki zalicza się także zagadnienia związane z projektowaniem systemów operacyjnych i oprogramowania.

Algorytmy i struktury danych

- typ danych to zbiór pewnych wartości i zbiór czynności na nich wykonywanych, jest on implementowany z wykorzystaniem jakiejś struktur danych (tworząc klasę w C++ wymyślamy typ danych)
- struktura danych - tablicowa i listowa oraz ich warianty
- problem algorytmiczny - zbiór danych, rozwiązań i relacja przekształcająca dane w rozwiązania (rozwiązania w tym zbiorze to zarówno rozwiązania dobre jak i złe)

- asymptotyczna mniejszość - $f(n)$ co najwyżej rzędu $g(n)$ - dla każdego $n > m$ zachodzi $f(n) < cg(n)$ - $f(n) = O(g(n)) \Leftrightarrow \lim_{n\to\inf} \frac{f(n)}{g(n)} = 0$
- asymptotyczna większość - $f(n)$ co najwyżej rzędu $g(n)$ - dla każdego $n > m$ zachodzi $f(n) > cg(n)$ - $f(n) = O(g(n)) \Leftrightarrow \lim_{n\to\inf} \frac{f(n)}{g(n)} = \inf$
- asymptotyczna dokładność - $f(n)$ co najwyżej rzędu $g(n)$ - dla każdego $n > m$ zachodzi $c_1g(n) < f(n) < c_2g(n)$ - $f(n) = O(g(n)) \Leftrightarrow \lim_{n\to\inf} \frac{f(n)}{g(n)} = G > 0$

Klasy złożoności:

- stała (1)
- logarytmiczna ($logn$)
- liniowa (n)
- liniowo logarytmiczna ($nlogn$)
- wielomianowa ($n^k, k = 2, 3, 4, ...$)
- pod-wykładnicza (n^{logn})
- wykładnicza (k^n)
- silniowa ($n!$)
- nad-wykładanicza (n^n)
- nieskończona

Zauważalny jest ogromny wzrost czasu dla algorytmów o złożoności gorszej niż wielomianowa wraz z wzrostem wielkości problemu

- problem klasy P - są deterministyczne algorytmy rozwiązujące go w czasie wielomianowym
- problem klasy NP - są nie deterministyczne algorytmy rozwiązujące go w czasie wielomianowym
- problem NP-zupełny - grupa problemów NP przekształcalnych jeden w drugi algorytmem wielomianowym
- problem silnie NP-zupełny - poszukiwanie lokalnego minimum jest problemem NP zupełnym
- problem NP-trudny - niedeterministyczne algorytmy rozwiązują go w czasie wykładniczym

metody:

- probabilistyczne - losowe wybieranie rozwiązania
- prób i błędów - losowanie (próba) i sprawdzanie wyniku (powtórzenie losowania x-razy)
- konstrukcyjne - budowanie rozwiązania w oparciu o jakieś przesłaniki
- optymalizacyjne - utworzenie rozwiązania i poprawianie go
- znalezienie wszystkich rozwiązań (przegląd bezpośredni) i znalezienie optymalnego (np. poprzez budowę drzewa)
- programowanie dynamiczne - jak poprzednio, ale z zauważeniem że do niektórych rozwiązań nie trzeba wielokrotnie dochodzić
- algorytmy genetyczne - naśladowanie ewolucji
- stymulowane odprężanie - naśladowanie krystalizacji (forma optymalizacji gdzie zgadzamy się na chwilowe pogarszanie rozwiązania z jakimś prawdopodobieństwem na skutek "drgań termicznych")
- systemy ekspercie - szczególny przypadek konstrukcyjnych

Algorytm wyszukiwania przez podział polega na dzieleniu posortowanej tablicy na części i próbie oszacowania w którym rejonie jest interesujący element

Algorytmy sortowania

Algorytm sortowania nazywamy **stabilnym** gdy dane o jednakowym kluczu sortowania nie zamieniają się miejscami.

- przez selekcję - wybieramy maksymalny, potem ze zbioru pomniejszonego o niego znowu (można zrobić w miejscu przenosząc na początek tablicy a znajdujący się tam na miejsce znalezionego)
- przez wstawianie - zbiór podzielony na część posortowaną i nie, pierwszy z nieposortowanej porównywany z ostatnim z posortowanej jeżeli mniejszy to zamieniany potem z następnym z posortowanej i tak do póki będzie nie mniejszy, część posortowana powiększana o jeden element i zajmujemy się kolejnym z nieposortowanej (można znaleźć pierwszy element posortowanej np. przez selekcję, ale aby zachować stabilność trzeba to zrobić sprytnie - zamiast zamiany przesunięcie)
- sortowanie Shella - ciąg kolejnych sortowań z malejącym krokiem (sortowanie z krokiem k - sortujemy dane tak aby pod-tablice utworzone przez wybór co co k-tej komórki były posortowane)
- sortowanie bombelkowe, zaczynamy od ostatniego, porównujemy z poprzednim i opcjonalnie zamieniamy (gdy nie zamieniamy przechodzimy do następnego elementu)
- quick-sort - metoda typu "dziel i rządź", dzielimy tablicę na dwie części (tak aby w jednej były tylko elementy mniejsze niż w drugiej) i każdą część sortujemy niezależnie, podziału dokonujemy przeglądając z obu stron i w zależności od wyniku porównania zamieniając elementy
- sortowanie o złożoności liniowej - dobre dla elementów o kluczu całkowitym i ograniczonej maksymalnej wartości klucza - tworzymy tabele krotności wystąpień kluczy (gdzie klucz jest indeksem), następnie w oparciu o te informacje budujemy tabelę danych posortowanych
- przez scalanie - łączenie dwóch posortowanych ciągów w ciąg posortowany
- sortowanie turniejowe (w tablicy) - porównujemy pierwszy z ostatnim, drugi z przedostatnim itd, potem powtarzamy na pierwszej części tablicy aż na początku znajdzie się największy (odpowiednio porównując w drugiej na końcu możemy umieścić najmniejszy)
- szybkie sortowanie binarne - złożoność nk dla k-bitowej liczby - sortujemy wg. najstarszej cyfry, wstępnie podzielone zbiory wg. olejnej itd
- sortowanie pozycyjne - gdy porównujemy nie liczby binarne a coś o większej podstawie

Grafy

Graf to zbiór wierzchołków V i krawędzi L.

- spójny - istnieje połączenie między dwoma dowolnymi wierzchołkami
- cykliczny - są drogi prowadzące od obecnego wierzchołka do niego bez powtórzeń krawędzi
- skierowany - krawędzie mają kierunek
- ważony - do krawędzi przypisane są wagi
- planarny - może zostać narysowany na płaszczyźnie tak aby krawędzie się nie przecinały
- droga w grafie - skończony ciąg węzłów i krawędzi, kończący i zaczynający węzłem bez powtórzeń krawędzi
- droga zamknięta - gdy wierzchołek startowy jest końcowym
- droga otwarta - gdy wierzchołek startowy nie jest końcowym
- ścieżka - droga otwarta, gdzie każdy wierzchołek występuje raz
- cykl (obwód) - droga zamknięta, gdzie każdy wierzchołek występuje raz
- droga Eurela - droga zamknięta przechodząca przez każdy wierzchołek bez przecinania krawędzi i powtórzeń krawędzi i wierzchołków
- obwód Hamiltona - obwód przechodzący przez każdy wierzchołek ale tylko raz

- dendryt (drzewo rozpinające) - drzewo powstałe z danego grafu po usunięciu pewnej liczby krawędzi
- MST - minimalne drzewo rozpinające - drzewo rozpinające o najmniejszej sumarycznej wadze krawędzi
- drzewo Stainera - MST dla grafu wzbogaconego o dodatkowe węzły tak aby zmniejszyć sumaryczne wagi połączeń
- minimalny przekrój - zbiór krawędzi o minimalnej sumarycznej wadzę po usunięciu których drzewo ulega podziałowi na dwa

Drzewa

Drzewem nazywamy graf acykliczny skierowany.

— przodek - wierzchołek z którego można przyjść do nas
— bezpośredni przodek (ojciec) - wierzchołek z którego można przyjść do nas pokonując jedną krawędź
— potomek - wierzchołek do którego można dojść od nas
— liście - wierzchołki z których nic nie wychodzi
— wysokość drzewa liczymy liczbą krawędzi od korzenia do najdalszego liścia
— głębokość wierzchołka - ilość krawędzi (odległość) od korzenia
— wysokość wierzchołka - wysokość drzewa minus głębokość wierzchołka
— zbiór rozłącznych drzew - las
— m-drzewo - drzewo uporządkowane gdzie każdy węzeł ma maksymalnie m bezpośrednich potomków (dzieci)
— drzewo zrównoważone - drzewo o minimalnej wysokości

— kopiec - drzewo pełne (wszystkie wiersze pełne prócz ostatniego, który musi być dosunięty do lewej) w którym każdy element większy lub równy swoim dzieciom
— drzewo poszukiwań binarnych po lewej mniejsze od danego wierzchołka a po prawej od niego większe, aby miało sens powinno być bliskie zrównoważonemu

Przechodzenie przez drzewa:

— preorder (VLP) - węzeł, lewe poddrzewo, prawe poddrzewo
— inorder (LVP) - lewe poddrzewo, węzeł, prawe poddrzewo
— postorder (LPV) - lewe poddrzewo, prawe poddrzewo, węzeł

Drzewa zrównoważone z definicji:

— 2-3-4 - węzły z dwoma, trzema lub czterema potomkami, węzły typu 4 rozbijane na 3 węzły typu drugiego
— czerwono-czarne, sprytnie zakodowane drzewo 2-3-4 gdzie są węzły tylko typu drugiego, ale łączone w grupę (kolorowanymi krawędziami) w ”wirtualne” węzły innych typów

Znajdowanie MST:

— algorytm Prima - z pośród wszystkich krawędzi wychodzących z drzewa dobieramy tą o najmniejszej wadze aż dobierzemy n-1 krawędzi (krawędzie wewnątrz drzewa nie stanowiące krawędzi drzewa usuwamy)
— algorytm Borůvki - podobny do Prima tyle że zaczynamy od wielu wierzchołków na raz - tworzymy las drzew poprzez dobranie dla każdego wierzchołka krawędzi o najmniejszej wadze
— algorytm Kruskala - sortujemy zbiór krawędzi, wybieramy najtańszą i usuwamy te które już nie wychodzą poza drzewo

Znajdowanie najkrótszej ścieżki w grafie:

— algorytm Dijkstry:

  0) każdemu wierzchołkowi dajemy wagę nieskończoną, a startowemu zero

  1) wierzchołek z najmniejszą wagą etykietuje się jako ostatni (znaleziony)

  2) jeżeli ostatni jest końcowym to kończymy

  3) dla wszystkich do tej pory nie zaetykietowanych oblicza się wagi wg zasady że jeżeli jest krawędź pomiędzy ostatnim a tym to nowa waga jest minimum z pośród obecnej i wagi ostatniego powiększonej o wagę tej krawędzi, nie radzi sobie z ujemnymi wagami i krawędziami wychodzącymi i wchodzącymi do tego samego wierzchołka
— Floyda - rozwinięcie Dijkstry na coś co znajduje wszystkie najkrótsze ścieżki: buduje się N macierzy (N = l. wierzchołków)
  $w_{ij}^{(0)}$ = {zawartość (w) komórki ij macierzy 0} = d_{ij} = odległość gdy bezpośrednie połączenie lub nieskończoność gdy go brak
  $w_{ij}^{(l)} = min(w_{ij}^{(l-1)}, w_{il}^{(l-1)} + w_{lj}^{(l-1)})$

  w praktyce wystarczy jedna macierz (bo stare nam nie potrzebne) i dodatkowa macierz w której zapisujemy wartość l (numer wierzchołka) gdy w + w < w

bisekcja grafu - Kernighnan-Lin - konstrukcja do wstępnego podziału:

— dla każdego węzła obliczamy koszt wewnętrzny (I) i zewnętrzny (E) - odpowiednio suma wag krawędzi pozostających w grupie i wychodzących do drugiej grupy
— dla każdej pary (po jednym węźle z grupy) liczymy zysk związany z zamianą danej pary ($D(v_i, v_j) = D(i) + D(j) - 2c(i, j)$, gdzie $D(i) = E(i) - I(i)$, $c(i, j)$ - waga krawędzi między i-tym a j-tym)

— parę z największym zyskiem usuwa się i ponawiamy procedurę
— ustalamy listę par wraz z wartościami zysku i znajdujemy miejsce gdzie kumulowany zysk jest największy i pary do tego miejsca zamieniamy
— całą procedurę powtarzamy aż nie będziemy uzyskiwać dodatniego zysku

Wprowadzenie do sieci neuronowych

Sieci neuronowe[1174] są systemami zdolnymi do przetwarzania danych w problemach trudno algorytmizowalnych. Cechują się odpornością na zniekształcenia danych, zdolnością do uczenia w oparciu o przykłady, zdolnością generalizacji (uogólniania). Dane przetwarzane są w sposób równoległy. Zobacz też: Sztuczna inteligencja[1175].

Budowa sieci wielowarstwowych nie ma sensu w przypadku sieci liniowych, czyli takich w których wektor odpowiedzi y jest liniowo zależy od wektora wejściowego x. Dla pojedynczego neuronu wygląda to w sposób następujący - wyjście jest sumą iloczynu wartości wejścia i wagi odpowiedniego wejścia:
$y^{(m)} = W^{(m)} * x = \sum_{i=1}^{n} w_i^{(m)} x_i$

Dzieje się tak dlatego że każde odwzorowanie liniowe możemy zapisać jako macierz, a wynikiem mnożenia dwóch macierzy przez siebie (co odpowiada kolejnemu wykonaniu pierwszej i drugiej warstwy) jest macierz (czyli pojedyncza warstwa). Uczenie neuronu polega na modyfikacji macierzy wag tak aby uzyskiwać lepszą odpowiedź. W przypadku uczenia z nauczycielem w oparciu o regułę delta chcemy minimalizować funkcję błędu w sensie najmniejszych kwadratów). Musimy zatem zmieniać wagi w kierunku przeciwnym do gradientu funkcji błędu:
$W' = W + \eta\delta X \qquad \delta = z - y$
$\frac{\partial Q}{\partial y} = -(z - y) = -\delta \qquad \frac{\partial y}{\partial W} = x \qquad \frac{\partial Q}{\partial W} = \frac{\partial Q}{\partial y}\frac{\partial y}{\partial W} = -\delta x_i$

Możliwe jest także uczenie bez nauczyciela (bez wiedzy o tym jaki powinien być wynik), w metodzie tej wzmocnieniu ulegają wagi dla których odpowiedź jest duża gdy sygnał pobudzający jest duży.

Celem możliwości tworzenia bardziej zaawansowanych odwzorowań wprowadza się elementy nieliniowe, w których wyjście jest funkcją ważonej sumy elementów wejściowych). Najprostszą realizacją nieliniowości jest $y = \{1, 0\}$ w zależności od znaku pobudzenia.Uczenie odbywa się na podobnej zasadzie jak w przypadku sieci liniowej. Jednowarstwowa sieć nie jest na przykład w stanie realizować odwzorowania XOR, jednak trój-warstwowa sieć (a kolejne warstwy mają sens tylko dla sieci nieliniowych) jest w stanie reprezentować dowolny podział płaszczyzny, zatem także funkcję XOR. Uczenie sieci wielowarstwowych odbywa się poprzez rzutowanie błędu na wszystkie neurony stanowiące wejście mnożąc go przez wagę danego wejścia. Należy zaznaczyć iż więcej neuronów w warstwach ukrytych daje więcej płaszczyzn decyzyjnych, jednak nie zawsze przyczynia się to do lepszego wyniku. W uczeniu sieci nieliniowych spotkać się możemy także z problemem lokalnych minimów do których będą zbiegały wektory wag (pomimo iż nie jest to właściwe rozwiązanie).

Celem usprawnienia procesu uczenia stosuje się różne metody dodatkowe takie jak:

— zanikanie wag - w każdym kolejnym kroku od wag odejmowany jest pewien czynnik
— bezwładność - wartość zmiany z poprzednich iluś kroków wpływa na wartość obecnej zmiany
— kontrola czy zmiana faktycznie przyczyniła się do zmniejszenia funkcji błędu jak nie to cofamy
— drgania termiczne - dodawanie losowej zmiany (zapobiega to wpadaniu w minima lokalne)
— stosowanie wstępnej obróbki danych
— stosowanie innych procedur minimalizacji

Szczególnym rodzajem są sieci przesyłające żetony (CP) - jest to dwuwarstwowa sieć, w której pierwsza warstwa klasyfikuje otrzymany wektor (z pierwszej warstwy wybieramy tylko wynik z jednego neuronu - największy), każdy neuron drugiej warstwy na swoim wejściu dostaje tylko jedyną liczbę i produkuje w oparciu o nią wektor skojarzony. Sieć taka działa jak tablica przeglądowa. Sieci tego typu stosowane są np. do kompresji danych, analizy statystycznej, rozpoznawania mowy, itp.

Kolejnym typem sieci są sieci RBF. Sieci te posiadają pola receptorowe, trafienie w które powoduje aktywacje danej jednostki wejściowej, oczywiście pola te mogą na siebie nachodzić wtedy pobudzeniu ulega kilka jednostek. Warstwa wejściowa dokonuje klasyfikacji bodźców, a wyjściowa wykonuje operację liniową. Sieci te charakteryzują się dobrą interpolacją[1176] lecz kiepską ekstrapolacją[1177].

Ostatnim omówionym tutaj rodzajem sieci są sieci Hopfielda. Są to sieci o połączeniach każdy z każdym (wektor wyjść podawany na wejścia). Sieci te mogą służyć do realizacji pamięci adresowanej treścią, do realizacji takiej koncepcji korzystamy z symetrycznych wag, wyjść mających wartości 1 i -1 i danych zależnością:
$S'_i = \mathrm{sgn}\left(\sum_j w_{ij} s_j\right)$

Aktualizacja stanów jest asynchroniczna i realizowana poprzez aktualizację (losowo wybrana jednostka dokonuje aktualizacji). Sieci te korzystają z funkcji energetycznej danej zależnością:
$H = -\frac{1}{2} \sum_{ij} w_{ij} s_i s_j$

Zobacz też: Logika rozmyta[1178], Artificial Neural Networks[1179].

Programy wspomagające kompilację

Budowanie większych projektów przy pomocy ręcznego kompilowania i linkowania poszczególnych elementów projektu byłoby udręką stąd bogactwo narzędzi automatyzujących ten proces - chyba nadal najpopularniejszym pozostaje make[1180]. Jednak jest on narzędziem dość ciężkim i m.in. dlatego powstały narzędzia go wspomagające (służące do budowania plików makefile) z grupy autotools[1181] (autoconf[1182], automake[1183], libtool[1184]) czy też CMake[1185]. Powstały także całkiem alternatywne narzędzia dla make takie jak bardzo interesujący SCons[1186] (zobacz też: dokumentacja SCons[1187]).

Systemy kontroli wersji

Narzędzia tego typu używane są głównie przez programistów, ale mogą się przydać także np. administratorom do wersjonowania konfiguracji.

SVN

Subversion[1188] (czyli svn) jest systemem kontroli wersji z centralnym repozytorium.

```
# pobranie repozytorium
svn checkout
# aktualizacja
svn update
# dodawanie i usuiwaniw plików
svn add|rm
# wysyłanie zmian a serwer
svn commit
```

Podstawowymi narzędziami są komendy `svn` (do zarządzania kopią roboczą) i `svnadmin` (do zarządzania repozytorium po stronie serwera. Kilka przykładów użycia:

- zmiana adresu URL repozytorium w kopii roboczej - `svn switch --relocate STARYNOWY`
- historia zmian pliku - `svn log PLIK`
- podgląd starszej wersji pliku svn `cat -r revisionPLIK`
- wykonanie zrutu z wybranych rewizji (np. tylko aktualnego) - `svnadmin dump katalog -r revision`
- import zrzutu (do pustego spowoduje zaimportowanie UUID) - `svnadmin load katalog`

URL może wskazywać dostęp np. w oparciu o ssh - `svn+ssh://host/katalog/repo`.

BZR

Bazaar[1189] (czyli bzr) jest rozproszonym systemem kontroli wersji mogącym jednak funkcjonować jako klasyczny system z centralnym repozytorium.

obsluga branchów

```
# utworzenie mirrora wskazanego branch'a
bzr branch
# aktualizacja (lub utworzenie) wskazanego mirrora obecnego brancha
bzr push
# zamiana bierzącego brancha w mirror wskazanego brancha (aktualizacja brancha)
bzr pull
# warto wspomnieć że:
# * podobny efekt jak pull daje merge+commit, przy czym zapewnia ono zachowanie
↪  hierarchii w historii zmian
# * branch'e mogą operować na pojedyńczym repozytorium (shared repository) - patrz bzr
↪  init_repo
```

OBSLUGA KOPII ROBOCZEJ

```
# utworzenie kopii roboczej
bzr checkout
# w ogólności możliwe jest tworzenie kopii z dala od brancha (np. na innej maszynie),
# w przypadku kożystania z checkout'ów działania komend takich jak update, commit
# odnoszą się do macierzystego brancha (działanie bzr przypomina wtedy działanie svn)
# możliwe jest także przełączanie macierzystego brancha dla checkotów (komenda switch)
# a także komitowanie lokalne (opcja --local w commit) lub nawet odłączenie i ponowne
# przyłączenie do brancha (unbind / bind) - w zasaadzie konwersjac checkout/branch
# scalanie zmian z innego brancha / mirrora brancha
bzr merge
# dodawanie usówanie i przenoszenie plików
bzr add|rm|mv
# akceptowanie zmian wprowadzonych kopii roboczej (do związanego z nią brancha)
# bzr commit -m "Opis zmian"
# aktualizacja kopii roboczej
bzr update
# cofanie kopii roboczej do stanu zgodnego z branchem
bzr revert
# warto wspomnieć że (w odróżnieniu od svn):
# * `bzr add *` w katalogu głównym bardzo ładnie dodaje wsystkie nowe obiekty
#    w całym drzwie podkatalogów (działa rekursywnie i po cichu ignoruje
#    obecne już w repo obiekty)
# * `bzr mv` przenosi obiekt w repozytorium a nie usuwa go i tworzy od nowa
#    (istotne przy zmianach nazw/położeń dużych plików)
```

ODZYSKIWANIE STARYCH WERSJI

```
bzr log
bzr -r $REVNUM ls
bzr -r $REVNUM cat
```

INNE

```
bzr diff
bzr status
bzr info
bzr export DST
bzr init
bzr whoami "Imie Nazwisko <adres@mailowy>"
```

GIT

Git[1190] jest rozproszonym systemem kontroli wersji.

UTWORZENIE PUSTEGO REPOZYTORIUM

```
mkdir -p NAME && cd NAME && git init
```

utworzenie kopii repozytorium (bez checkout-u / bez katalogu roboczego)

```
# metoda wypychająca
#  chyba nie istnieje
# metoda wciągająca
git clone -n ssh://user@host/sciezka/do/repozytoriumNEWREPO
```

aktualizacja kopii repozytorium

```
# metoda wypychająca
git push ssh://user@host/sciezka/do/repozytorium
# metoda wciągająca
#  -> tworzy katalog roboczy
#  -> w pewnych sytuacjach robi jawnego merge
#  -> automatycznie commituje jeżeli nie ma konfliktów
#  -> pozwala na połączenie dwóch zupełnie niezależnych repozytoriów w jedno
#     (niestety robi to bez wymagania jakiegokolwiek potwierdzenia)
git pull ssh://user@host/sciezka/do/repozytorium master
```

Możemy też ustawić automatyczne robienie push po wykonaniu commit poprzez utworzenie stosownego pliku `.git/hooks/post-commit` (po stronie robiącej push), plik musi mieć atrybut wykonalności. Jeżeli chcemy żeby push działało do repozytorium na którym mamy utworzoną kopie roboczą w `.git/config` (po stronie przyjmującej push) należy ustawić:

```
[receive]
        denyCurrentBranch = warn
```

Jeżeli mamy poprawnie zdefiniowane w `.git/config` wpisy: `[remote "origin"]` i `[branch "master"]` (jest tak np. gdy nasze repo powstało poprzez "clone" repo z którego chcemy robić pull) to zamiast powyżej pokazanej pełnego wywołania pull możemy używać krótszej (i bezpieczniejszej - bez ryzyka wciągnięcia innego repozytorium) bezargumentowej postaci: `git pull`.

obsługa katalogu roboczego w oparciu o lokalne repozytorium

```
# aktualizacja
git checkout -f
# zatwierdzanie zmian (wszystkich)
git add .; git commit -a -m "OPIS"
# uzytkowanie
git mv
git status  # aby ignorować jakieś niedodane do repozytorium pliki katalogi należy
 ↪  skorzystać z pliku .gitignore
git revert -n 42200238cbd4c862a0b6ce919e5a4dc2a47cc20b
git ls-tree REV
git cat-file blob file-SHA1
git show rev:path
```

Debugowanie (gdb)

GDB czyli GNU Debugger jest narzędziem służącym do podglądania działania innego programu. Umożliwia ustawianie pułapek (powodujących przerwanie wykonywania na wywołaniu określonej funkcji, uruchamianie programu linia po linii, podgląd wartości zmiennych, … . Umożliwia także sprawdzenie co program robił gdy się wywalił (na podstawie pliku core). Jest to narzędzie programistyczne, które niewątpliwie warto poznać, gdyż znacznie ułatwia poprawianie błędów w programach. Na wstępie polecam lekturę `man  gdb`, `info  gdb`.

Dla skorzystania z części możliwości konieczne jest wkompilowanie informacji debugerowych w program (np. gcc z opcją -ggdb). Program uruchamiany jest przez wywołanie gdb, zazwyczaj z jedną lub dwiema opcjami - plikiem wykonywalny oraz (ewentualnie) plikiem core. Do najistotniejszych poleceń należy:

`break nazwa_funkcji` ustawiające pułapkę na daną funkcję,
`run` uruchamiające program (zatrzymuje się na pierwszej napotkanej pułapce),
`continue` powodujące kontynuację programu do następnej pułapki,
`next` powodujące wykonanie kolejnej linii programu,
`nexti` powodujące wykonanie kolejnej instrukcji procesora,
`print nazwa_zmiennej` powodujące wypisanie wartości podanej zmiennej,
`help` wyświetlające pomoc ”on-line”,
`quit` kończące pracę gdb.

Oczywiście to tylko niewielka część wszystkich poleceń gdb, polecam zapoznać się z poszczególnymi grupami poleceń opisywanymi przez `help`.

GDB umożliwia również debugowanie programów na zdalnych systemach (podłączonych przez port szeregowy bądź przez sieć - zobacz `man gdbserver`), możliwość komunikacji przez sieć wykorzystują też programy zewnętrzne (takie jak simulavr, ice-gdb, ...).

Uruchamianie poprzez GDB przydaje się także przy debugowaniu problemów typu ”segmentation fault” - po zatrzymaniu programu z powodu takiego błędu możemy wydać w gdb polecenie ”backtrace” aby zobaczyć w której funkcji wystąpił problem

Wspomniany simulavr jest symulatorem mikrokontrolerów AVR (niestety nie działającym jeszcze idealnie ...) i z przystosowaną dla programowania AVR w C wersją gdb (avr-gdb), umożliwia stworzenie systemu do testowania i debugowania programów AVR na zwykłym komputerze. Aby programy te zechciały ze sobą współpracować należy:

Uruchomić simulavr z opcjami -g i -p oraz określeniem odpowiedniego urządzenia i pliku z skompilowanym programem, np. (dla atmega128): `simulavr -d atmega128 -g -D -P simulavr-disp -p 2345 moj_program.hex`.

Uruchomić avr-gdb z określeniem pliku zawierającego skompilowany program, a następnie kazać połączyć mu się z simulavr, poprzez komendę: `target remote localhost:2345` (numer portu oczywiście możemy zmienić, ale w obu programach musi być ten sam).

Zamiast simulavr możemy sprobować debugowania w układzie za pomocą avarice: `avarice -d -f moj_program.hex -P atmega128 --jtag /dev/ttyS0 :2345`, natomiast jeżeli chcemy obejrzeć komendy asemblerowe zawarte w wygenerowanym kodzie polecam (analogicznie do objdump dla zwykłych programów): `avr-objdump -d moj_program.elf` lub (specjalnie dla AVR): `revava -e moj_program.hex`

Zobacz w Sieci: GCC Debugging[1191].

The GNU Profiler

Często gdy zachodzi potrzeba przyspieszenia działania jakiejś aplikacji, programista staje przed trudnym zadaniem ustalenia co w niej chodzi zbyt wolno. W zadaniu tym nieocenioną pomocą są programy profilujące. Z generowanego w trakcie pracy programu logu profilowania wyciągają informacje o liczbie wykonania poszczególnych funkcji czy też nawet pojedynczych linijek kodu. Takim właśnie programem jest gprof.

Biblioteki .so i .a

Przy problemach z linkowaniem przydatne może być wyświetlenie symboli które zawiera konkretna biblioteka. Możemy to zrobić poleceniami:

```
#listowanie symboli w pliku .so:
        nm -g       PLIK.so
        nm -gC      PLIK.so  # for C++ libs
        readelf -Ws PLIK.so  # for ELF libs
#listowanie symboli w pliku .a:
        nm          PLIK.a
```

Qt-Designer

Qt-Designer (`designer`) jest programem służącym do wizualnego tworzenia graficznego interfejsu opartego na bibliotece Qt. Program ten operuje na plikach xmlowych które na kod źródłowy przetwarza się poleceniem `uic nazwapliku.ui`. Natomiast celem zbudowania programu wystarczy wydać kolejno `qmake` (generuje on plik Makefile dla całego projektu qt - plików *.ui, *.ui.h) oraz `make` (kompilacja i linkowanie w oparciu o wygenerowany wcześniej plik Makefile). Program `qmake` wymaga pliku opisującego projekt (potrafi go wygenerować sam Qt-Designer, jednak można posłużyć się także samym `qmake` z parametrem `-project` (w projekcie uwzględnia wszystkie pliki o odpowiednich rozszerzeniach z bieżącego katalogu)).

Qt3

Niestety Qt-Designer nie udostępnia (a przynajmniej ja nie znalazłem) możliwości wydania tych poleceń z niego, ani nie generuje funkcji main, niezbędnej do uruchomienia przygotowanego w nim program (GUI wraz z kodem akcji, który możemy w nim edytować). Poniżej przedstawiam szablon funkcji main dla głównego okna i klasy z nim związanej o nazwie `MainWindow`, kod ten należy umieścić np. w swoim pliku *.ui.h :

```
#include <qapplication.h>
int main( int argc, char **argv ) {
         // tworzymy obiekt zarządzający zasobami aplikacji
        QApplication apps( argc, argv );
         // utworzenie obiektu z klasy moje okno
        MainWindow main_window;
         // ustalamy że pudełko box0 będzie głównym oknem aplikacji
        apps.setMainWidget( &main_window );
         // pokazujemy nasze okno ...
        main_window.show();
        // pętla główna programu i zwrócenie kodu zwróconego przez a.exec() do rodzica
 ↪  ...
        return apps.exec();
}
```

Qt4

Zobacz w dziele o c++.

Grafika 3D i gry

W programowaniu współczesnych gier[1192] bardzo istotnym elementem jest obsługa grafiki (zazwyczaj 3D aczkolwiek nadal można spotkać i zapewne zawsze będzie liczna grupa gier 2D). Rzadko graficzna część gry tworzona jest od podstaw z bezpośrednim wykorzystaniem mechanizmów takich jak OpenGL[1193]. Najczęściej gry korzystają z silników gier[1194] zawierających silniki graficzne[1195] lub bezpośrednio z tych drugich.

Silnik gry różni się od silnika graficznego tym że dostarcza także mechanizmów służących do implementacji fizyki świata gry (w tym wykrywania kolizji[1196]), sztucznej inteligencji[1197], obsługi wejścia (sterowania), dźwięku i sieci. Silnik graficzny dostarcza natomiast tylko wygodnych mechanizmów operowania obiektami graficznymi, maskując niżej poziomowe interfejsy takie jak SDL[1198], czy jeszcze bardziej niskopoziomowy i ograniczony tylko do grafiki OpenGL. Wspomniany OpenGL jest API do obsługi grafiki 3D, najczęściej stosowany jest w zastosowaniach profesjonalnych, ale niekiedy także w grach - ze względu na otwartość specyfikacji oraz dostępność na wiele platform - zwłaszcza w grach z rodziny FLOSS[1199]). SDL jest z kolei biblioteką programistyczną zwalniającą programistę z konieczności operowania suchym OpenGL i jemu podobnymi mechanizmami, jednak jest dość niskopoziomowy i nie jest określany mianem silnika graficznego czy też silnika gry (dostarcza jednak mechanizmów obsługi wejścia itp).

Jednym z głównych zadań silnika graficznego jest robienie cullingu[1200] celem ograniczenia ilości geometrii ładowanej do potoku graficznego, pociąga to za sobą mechanizmy zarządzania przestrzennego sceną. Ponadto silniki muszą zarządzać samymi obiektami, ich ładowaniem oraz ustawianiem wymaganych przez nie parametrów potoku (w tym ładowaniem wymaganych shaderów[1201]). Z kolei podstawowym zadaniem silników fizycznych jest realizacja

wykrywania kolizji. Na ogół do obu tych mechanizmów korzysta się z brył ograniczających. W grafice 3D często wykorzystuje się tekstury do innych niż standardowe zastosowań (np. przechowywania map oświetlenia).

Do najbardziej zaawansowanych projektów silników na wolnych licencjach należą: Irrlicht[1202] (licencja typu BSD), OGRE[1203] (LGPL), Crystal Space[1204] (LGPL). Dość bogatą bazę silników (w tym tych z rodziny FLOSS) można przeglądać na devmaster[1205].

Interesującą koncepcją wydaje się stosowanie zwykłej kamery celem tworzenia modeli 3d (skanowania 3d) - informacje na temata projektu - ProFORMA by Qi Pan @ Cambridge University[1206].

Zobacz też: porady dotyczące edytora grafiki 3D blender, przykładowe programy multimedialne w C/C++ (w tym związane z grafiką 3D), podstawa silnika 3d, portal o programowaniu gier[1207], edytor shaderów dla KDE[1208] (łatka[1209] ułatwiająca kompilację w Debianie).

Linki

Zobacz w Wikipedii: Programowanie komputerów @ Wikibooks[1210], x86 Disassembly[1211], biblioteka[1212], algorytmy[1213] (Algorithms @ Wikibooks[1214], Kategoria:Algorytmy[1215], Kategoria:Algorytmy_kompresji[1216], Algorithm Implementation[1217]), struktury danych[1218] (Kategoria:Struktury_danych[1219],abstrakcyjny typ danych[1220], b-drzewo[1221], drzewo[1222], drzewo AVL[1223], drzewo czerwono-czarne[1224], drzewo poszukiwań binarnych[1225], kolejka[1226], kolejka priorytetowa[1227], kopiec[1228], lista[1229], rekord[1230], stos[1231], tablica[1232], tablica asocjacyjna[1233], tablica mieszająca[1234]), sortowanie[1235] (sortowanie bąbelkowe[1236], sortowanie przez wstawianie[1237], sortowanie przez scalanie[1238], sortowanie przez zliczanie[1239], sortowanie kubełkowe[1240], sortowanie pozycyjne[1241], sortowanie przez wybieranie[1242], sortowanie Shella[1243], sortowanie grzebieniowe[1244], sortowanie szybkie[1245], sortowanie introspektywne[1246], sortowanie przez kopcowanie[1247]), metody_numeryczne[1248], [1249], Data Coding Theory[1250], Data Compression[1251], Digital Signal Processing[1252], Floating Point[1253], The Computer Revolution[1254], Communication Networks[1255], Computer Systems Engineering[1256], Operating System Design[1257].

Zobacz w Sieci: sterowanie z wykorzystaniem komputera[1258], 4programmers[1259], programowanie wg Stelmika[1260], Koders - Source Code Search Engine[1261], HotScripts.com[1262], A Short Tutorial on XMLHttpRequest()[1263] (czyli jak strona WW może robić zapytania HTTP).

Systemy oraz instalacje kablowe inteligentnego budynku

Mamy do czynienia z dwoma typami okablowania - instalacje zasilające (na ogół 230V/400V AC[1264], niekiedy także niskonapięciowe) oraz instalacje teletechniczne/teleinformatyczne (sieci komputerowe, CCTV, systemy sterujące, alarmowe, ...).

W przypadku instalacji zasilających kwestia okablowania jest prosta. Jako obwody rozprowadzające stosuje się najczęściej przewody o przekroju 1.5mm - instalacje oświetleniowe, 2.5mm - instalacje gniazd oraz grubsze (typowo 4 lub 6mm) dla odbiorników większej mocy. Jako główne linie zasilające wykorzystuje się większe przekroje. Przewody wysokonapięciowe (230VAC) mogą być bez żadnych problemów wykorzystane do instalacji niskonapięciowych (np. 12VDC[1265] lub 24VDC).

Natomiast w przypadku przewodów sygnałowych sytuacja (nadal) jest trudniejsza - mamy linie niesymetryczne (przewód współosiowy), gdzie ochroną przed zakłóceniami jest ekranowanie i symetryczne (skrętka), gdzie najczęstszą ochroną przed zakłóceniami jest transmisja różnicowa, a także różne impedancje tych linii (najczęściej 75 i 100 omów). Najpowszechniej obecnie używanym jest okablowanie symetryczne 100 omowe kategorii 5e lub wyższej (ze złączami RJ45[1266]) - jest ono wykorzystywane przede wszystkim do sieci komputerowych (Ethernet), ale może być również wykorzystane do przesyłu innych sygnałów (instalacje alarmowe, telefonia, systemy sterujące, ...). Za wykorzystywaniem go w innych instalacjach niż sieci komputerowe przemawia ujednolicenie okablowania, przeciw argumenty ekonomiczne (jest ono lepsze ale i droższe od tego jakiego wymagają te systemy). Okablowanie to może być także wykorzystywane do przesyłu analogowego sygnału audio i wideo (AV), ale tutaj problemem jest dopasowanie systemów AV do korzystania z tego medium, systemy te dostosowane są do korzystania z niesymetrycznego kabla 75 omów (ze złączami F[1267], BNC[1268] lub chinch (RCA[1269])) i konieczne jest użycie specjalnych układów dostosowujących. Są to układy oparte o indukcję (balun/transformator wideo) lub też specjalizowane układy elektroniczne (wzmacniacze operacyjne typu MAX435/MAX436 czy też LT1190/1193, które umożliwiają przesył obrazu z wykorzystaniem skrętki nawet na odległości większe niż 1km). Na rynku dostępne są baluny pozwalające także na łatwe przesyłanie zasilania i audio tą samą skrętką. Dla takiego systemu transmisji danych z kamer i zasilania tych urządzeń warto rozważyć zastosowanie przełącznicy wideo, która realizowałaby funkcję centralnego zasilania wszystkich kamer, komasowała sygnały z par kamer w jednym złączu RJ-45 (możliwe po wyeliminowaniu zasilania) oraz posiadała obok wyprowadzeń zwykłych wyprowadzenia których przyporządkowanie do danej kamery ustalane jest poprzez RS485.

Topologia instalacji

Topologia[1270] fizyczna instalacji jest to zbiór zasad według których rozprowadzana jest dana instalacja, jej okablowanie. Do najważniejszych modeli topologii okablowania należy zaliczyć:

- magistralę[1271] - wszystkie urządzenia są podłączone do jednej szyny danych, okablowanie nie wyróżnia punktu centralnego
- pierścień[1272] - magistrala której końce są połączone, uodparnia to na pojedyncze uszkodzenie
- gwiazdę[1273] - wszystkie podłączenia (punkt-punkt) początek biorą w węźle centralnym
- topologie mieszane - rozgałęzionej gwiazdy[1274], hierarchiczna[1275] (drzewa, rozproszonej gwiazdy), itp.; ogólnie można je podzielić na:
  - rozwiązania w których mamy do czynienia z gwiazdami łączonymi przy pomocy magistral, pierścieni lub innych gwiazd
  - rozwiązania w których mamy do czynienia z magistralami łączonymi przy pomocy gwiazd

Każda z wspomnianych topologii ma swoje zalety i wady. Z punktu widzenia zarządzania infrastrukturą, awaryjności, itd. najlepszą wydaje się topologia gwiazdy (łączy punkt-punkt do poszczególnych urządzeń), natomiast z punktu widzenia kablowania optymalnym rozwiązaniem jest najczęściej magistrala. Ponadto specyfika wielu z interfejsów określa to jakiej topologi należy użyć budując sieć opartą na tym interfejsie - np. i2c, rs485 wymagają magistrali.

W praktyce największe zastosowanie mają topologie mieszane - pozwalają one korzystać z zalet topologi gwieździstej, jednocześnie ograniczając ilość okablowania (poprzez podział jednej gigantycznej gwiazdy na wiele mniejszych). Często ich zastosowanie jest pożądane także w przypadku systemów dla których naturalna wydaje się magistrala, czy nawet jest ona wymagana przez standard - topologie mieszane możemy wtedy wykorzystać do

podziału jednaj magistrali na kilka zmniejszając tym samym jej obciążenie. W przytoczonym przykładzie i2c możliwe jest stosowanie switchy do podziału magistrali co umożliwia nam podpięcie większej ilości urządzeń, zamieniając jednocześnie topologię magistrali w topologię mieszaną.

Po mimo tradycyjnej naturalności realizacji instalacji elektrycznej w postaci magistral oraz znacznych ilości kabli wymaganych przy realizacjach gwieździstych końcowy odcinek (od ostatniej rozdzieli do urządzeń) w miarę możliwości warto (ze względu na wygodę sterowania) realizować w postaci gwiazdy. Natomiast na głównych liniach zasilających rozbicie typowej dla nich magistrali na kilka magistral wychodzących z rozdzielni umożliwia stosowanie mniejszych przekrojów (szyno-)przewodów.

W przypadku różnych instalacji hydraulicznych / rurowych często ciężko sobie wyobrazić coś innego niż magistrala, ale mimo to warto tutaj także w końcowym odcinku zastosować gwiazdę - umożliwi to łatwiejsze sterowanie i monitoring takiej instalacji, a także ułatwi ewentualne drobne poprawki (łatwy dostęp do połączeń rur które często znajdują się w ścianach lub podłogach).

Sposoby prowadzenia instalacji

Instalacje kablowe (z stosunkowo niewielką liczbą przewodów) mogą być prowadzone na kilka sposobów:

- w ścianach lub podłogach:
  - bezpośrednio (instalacje wtynkowe) - najprostszy do realizacji w trakcie budowy obiektu, lecz nie dający praktycznie żadnych możliwości ingerencji w instalację
  - w rurkach lub peszlach (instalacje podtynkowe) - dzięki stosowaniu rurek i puszek łączeniowych oraz rewizyjnych pozwalają na pewną ograniczoną ingerencję w instalację bez prac budowlanych, jednak rozbudowa takiej instalacji może być uciążliwa
  - korytkach kablowych do montażu podtynkowego (pokrywa koryta zlicowana z powierzchnią ściany) lub kanałach podłogowych (dostęp poprzez puszki podłogowe) - pozwala na dość nieuciążliwą rozbudowę i modyfikację instalacji
- na ścianach lub sufitach
  - bezpośrednio - prowadzenie gołego kabla mocowanego specjalnymi zaczepami - najprostszy do realizacji, umożliwia łatwe modyfikacje instalacji, jednak nie zapewnia żadnego ich uporządkowania i zabezpieczenia
  - w rurkach - rurki i akcesoria łączeniowe (złączki, złączki karbowane, kolanka, rozgałęźniki) typowo wykonane z PCV i mocowane do podłoża specjalnymi uchwytami
  - korytkach kablowych - typowo wykonanych z PCV, ale występują także systemy lakierowanych koryt metalowych - pozwala na łatwą rozbudowę i modyfikację instalacji

Na rynku dostępnych jest wiele systemów korytek kablowych (niestety niekompatybilnych ze sobą) umożliwiających budowę kompleksowych tras kablowych. Na systemy te oprócz samych kanałów instalacyjnych o różnych rozmiarach składają się także akcesoria takie jak: przegrody (zwykłe, umożliwiające montaż niezależnych pokryw, zapewniające powiększony odstęp, niekiedy są one częścią samego korytka), pokrywy (perforowane, pełne, zaczepiane od góry lub od boku kanału), różnego rodzaju łączniki, kolanka, narożniki, rozgałęźniki, reduktory rozmiarów, pokrywy elastyczne, klamry rozporowe, … . Kanały posiadają wykonania podsufitowe/narożnikowe, przypodłogowe, elementy umożliwiające montaż osprzętu - zarówno standardowego, jak też modułowego typu S (18mm) czy też 45x45mm ("EuroMod"), a także urządzeń takich jak czujniki, lampy, itp, zarówno na kanale jak i obok niego. W przypadku kanałów podłogowych wybór systemów także jest spory i praktycznie każdy z nich zawiera wiele akcesoriów (różnego rodzaju puszki przyłączeniowe, kolumny elektroinstalacyjne, itd). Powszechne jest tez wykorzystywanie sufitów podwieszanych[1276] (zwłaszcza złożonych z szkieletu wypełnianego standardowymi płytkami) do utworzenia przestrzeni na prowadzenie instalacji i ich jednoczesnego zamaskowania.

Większe instalacje kablowe (ale nie ma przeciwwskazań aby tymi rozwiązaniami posiłkować się także przy mniejszych instalacjach) możemy prowadzić z wykorzystaniem koryt lub drabin kablowych mocowanych przy użyciu konstrukcji wsporczych do ścian i sufitów. Zaleta tego rozwiązania jest także to iż kable typowo leżą na korycie lub drabinie - nie występuje problem wysypujących się z korytka kabli. W przypadku większych pomieszczeń technicznych bardzo użyteczna może być podniesiona podłoga techniczna[1277], umożliwiająca łatwą dystrybucję dużej ilości kabli (i/lub tras kablowych budowanych z korytek kablowych) po znacznej powierzchni. Koryta kablowe pod podłoga techniczną nie są konieczne (zwłaszcza dla obwodów dedykowanych do konkretnego urządzenia, bądź "dużych" kabli), ale warto je stosować przede wszystkim dla obwodów ogólnego przeznaczenia - optymalnym miejscem dla prowadzenia takich tras kablowych wydają się być okolice ścian danego pomieszczenia. Warto także zwrócić uwagę na dodatkowe ekranowanie elektromagnetycznego jakie zapewniają koryta i kanały metalowe.

Jako podstawę dystrybucji należałoby przyjąć system drabin i perforowanych koryt stalowych (które mogą być układane zarówno na widoku jak i mogą być maskowane podłogą techniczną/sufitem podwieszanym). Natomiast

na dalszych odcinkach warto zastosować metalowe kanały instalacyjne umożliwiające montaż osprzętu, uzupełnione kanałami podsufitowymi i mniejszymi listwami rozprowadzającymi pojedyncze obwody (np. do czujek). W pomieszczeniach technicznych bardzo praktyczne wydaje się uzupełnienie takiego systemu o korytka siatkowe (zapewniają łatwe wyprowadzanie przewodów, a także możliwość łatwego uzyskiwania łuków, redukcji, trójników itp bez korzystania z elementów prefabrykowanych - łatwe wycinanie z koryta podstawowego). W przypadku pomieszczeń technicznych istotne są bogate systemy mocowania ściennego i sufitowego (producenci oferują zazwyczaj całe konstrukcje nośne dla koryt, mocowane do sufitu czy nawet gołych belek, tak więc nie jesteśmy ograniczeni tylko do mocowania z wykorzystaniem prętów gwintowanych i tulejek rozprężnych z gwintem wewnętrznym / kołków rozporowych do śrub z gwintem metrycznym / dwugwintowej śruby (metryczny i wkręt) wraz z tulejką łączącą).

Dystrybucja najczęściej (obiekty biurowe, mieszkania, ...) ma charakter poziomy i dotyczy głównie obwodów między ostatnim punktem dystrybucyjnym a punktem końcowym instalacji (gniazdo, lampa, ...). Jednym z wyjątków (obok łączenia piętrowych punktów dystrybucyjnych w biurowcach) jest przypadek wielorodzinnego budynku mieszkalnego, gdzie dystrybucja mediów do lokali ma głównie charakter pionowy (niewielka powierzchniowo klatka schodowa z kilkoma lokalami na piętrze i sporo pięter). W takim wypadku najrozsądniejsze wydaje się prowadzenie dwóch pionów kablowych i łączenie ich podsufitową listwą (przechodzącą nad wejściami do wszystkich lokali), z której prowadzone są przepusty do wnętrza lokali i w której umieszczane są punkty styku instalacji wspólnej i wewnętrznej. Także w tej listwie (obok lokowanych tam gniazd styku instalacji wspólnej i wewnętrznej) należy zostawiać niewielki odcinek kabla jako zapas, podobnie jak przy każdym innym gniazdku końcowym.

Galeria zdjęć związanych z dystrybucją okablowania[1278]. Zobacz w Sieci: korytka siatkowe[1279], katalog koryt stalowych BAKS[1280], Łukasiak - Podłogi podniesione • Podłogi podwójne modułowe[1281], PODŁOGA TECHNICZNA PODNIESIONA MSL-100 metalowa[1282].

Punkty dystrybucyjne

Pod pojęciem punktu dystrybucyjnego będę rozumiał miejsce w którym zlokalizowane są punkty centralne instalacji realizowanych w topologii gwiazdy, znajdują się urządzenia rozgałęziające magistrale czy też nimi zarządzające. Typowo punkty takie mieszczą panele krosownicze[1283] (RJ45[1284], światłowodowe, F[1285], BNC[1286], ...), rozdzielnice elektryczne[1287], urządzenia sterujące, osprzęt sieciowy, a nawet jakieś drobne serwery. W zależności od stopnia rozbudowania instalacji mogą to być szafki krosownicze powieszone gdzieś pod sufitem, albo całe dedykowane pomieszczenia. Podobnie poszczególne instalacje mogą mieć niezależne punkty dystrybucyjne, lub też różne instalacje mogą korzystać z wspólnych punktów dystrybucyjnych (co ma pewne zalety). Miejsca takie pełnia bardzo ważną, wręcz strategiczną rolę w strukturze okablowania, systemów sterujących, sieci komputerowych i innych sieci teleinformatycznych.

Koncepcja realizacji[1288] niewielkiego (jedno - dwa pomieszczenia) punktu dystrybucyjnego w postaci szafki z stelażem rack 19” o wysokości 6U lub 10U.

Systemy rackowe 19-sto calowe

Najpopularniejszą formą realizacji telekomunikacyjnych punktów dystrybucyjnych są szafki lub szafy krosownicze wykonywane w standardzie rack[1289] 19” (w przypadku mniejszych sieci można się spotkać z także z standardem 10 calowym). Jest to standard[1290], który pozwala na montaż w jednej obudowie (szafie bądź stojaku) różnorodnego osprzętu, począwszy od paneli krosowych, osprzętu sieciowego, poprzez komputery przemysłowe, serwery i specjalistyczne urządzenia sterujące ..., po modułową aparaturę elektryczną. Urządzenia w tym standardzie charakteryzują się znormalizowaną szerokością (właśnie 19 cali czyli koło 48 cm) oraz wysokością w wielokrotnościach 1U[1291] (= 44,45 mm), nie posiadają natomiast zestandaryzowanej głębokości (na ogół nie przekracza ona 45 cm, ale potrafi być nawet 60 cm do 100 cm).

Ważną cechą **szaf** jest demontowalność boków szafy oraz przednie i tylne otwierane drzwi (co daje możliwość dostępu do urządzeń z dowolnej strony), producenci zapewniają też szeroki wybór rodzajów drzwi. Istnieje także możliwość zastosowania skróconych drzwi umożliwiające montaż na górze lub dole specjalnych przepustów kablowych, takowe mogą być też montowane w podłodze jak i suficie szafy (tam mogą być montowane też systemy wentylacji). W śród cech szaf wyróżnia się również spora gama szerokości (większa szerokość zapewnia większą przestrzeń z boku urządzeń, a także możliwość montażu ram obrotowych) Trzeba też wspomnieć o mniejszych od typowych stojących szaf wersjach przystosowanych do powieszenia na ścianie (tu warto zwrócić uwagę na szafy dzielone, które umożliwiają otwarcie ”od tyłu” - przednia część połączona jest z tylną mocowaną do ściany zawiasami).

Jeżeli mamy do dyspozycji całe dobrze zabezpieczone pomieszczenie warto rozważyć użycie zamiast szaf 19”

odpowiednich **stojaków** (jak szafa tyle że bez obudowy) - zapewnia to łatwiejszy dostęp do zamontowanych urządzeń. W przypadku gdy lokalizacja szafy uniemożliwia dostęp od tyłu warte rozważenia jest zastosowanie szaf z uchylną ramą montażową co umożliwia uzyskanie pełnego dostępu do zainstalowanych urządzeń poprzez obrócenie całej konstrukcji montażowej wraz z urządzeniami, natomiast w przypadku mniejszych (wieszanych na ścianie szafek) konstrukcji dzielonych. W przypadku szaf przeznaczonych do montażu serwerów wraz z osprzętem warto rozważyć szafy wyposażone w kółka jezdne, w połączeniu z wyposażeniem każdej takiej szafy w własne listwy PDU lub panel dystrybucji napięć (od przodu szyna do montażu zabezpieczeń, od tyłu gniazda podłączeniowe) oraz switch (co redukuje liczbę kabli dochodzących do szafy) ułatwia to znacznie reorganizację systemu.

W śród bogatego osprzętu wspomnieć należy o:

- panelach krosowych
- osprzęcie sieciowym (routery, switch'e, …), telefonicznym, …
- obudowach komputerów przemysłowych (zarówno opartych o zwykłe płyty główne, jak i specjalistyczne rozwiązania) o zróżnicowanej wysokości (najczęściej od 1U do 6U) i gamie elementów dostępnych z przodu (np: wyprowadzenia kart rozszerzeń oraz pojedynczych urządzeń 3.5" i 5.25" albo wyprowadzenia wyłącznie wielu urządzeń 5.25", 3.5" lub 2.5", …) oraz obudowach stacji roboczych z wbudowanym miejscem na wyświetlacz ciekłokrystaliczny
- obudowach i półkach do innego osprzętu, w tym konstrukcji specjalnie przystosowanych do urządzeń danego typu (np. monitorów), osprzęcie porządkującym kable
- wysuwanych szufladach na klawiaturę, dokumentację, …
- panelach dystrybucji napięć - umożliwiających montaż standardowego elektrycznego osprzętu modułowego na wspornik 35mm (szynę DIN[1292]) - do 18 modułów 17,5 mm przy wysokości 3U
- listwach zasilających PDU - także tzw. 0U (czyli nie zajmujących miejsca w głównej przestrzeni a montowanych gdzieś z boku), oraz tzw. inteligentnych (umożliwiających pomiar prądu, a nawet załączanie i wyłączanie poszczególnych gniazd)
- zasilaczach awaryjnych oraz innym specjalistycznym osprzęcie

Galeria zdjęć związanych z szafami rack'owymi[1293]

Rozdzielnice elektryczne

Główną z instalacji, której punkty dystrybucyjne umieszczane są zazwyczaj poza szafami dystrybucyjnymi typu rack jest instalacja elektryczna - zasadnieniem takiego postępowania są specyficzne wymogi rozdzielic elektrycznych oraz oszczędność przestrzeni. W przypadku dużych rozdzielnic niskiego napięcia na prądy rzędu kilku tysięcy stosuje się rozdzielnie wykonywane pod zamówienie albo modułowe systemy rozdzielnic (np. Schneider Prisma Plus System P do 3200A) w wielu aspektach (także pod względem gabarytowym) przypominających szafy rackowe. W przypadku prądów wynoszących kilkaset amperów najczęsciej stosuje się mocowane do ściany obudowy o głębokości 250cm - tutaj także można wybierać pomiędzy realizacją na zamówienie a systemami modułowymi wielu producentów (np. Hager Orion+ do 630A albo Schneider Prisma Plus System G do 160A). Przy jeszcze mniejszych prądach często (ale raczej nie w poważnych zastosowaniach przemysłowych) spotykane są płytkie (10cm) rozdzielnie z kilkoma rzędami szyny DIN.

Aparatura zabezpieczająca, sterująca i pomocnicza małych prądów (do około 125A) jest zestandaryzowana w postaci modułów do montarzu na szynie DIN[1294]/TS-35 (tzw. aparatura modułowa). Niestety standaryzacja ta dotyczy tylko ogólnych wymiarów i próba łączenia szynami grzebieniowymi aparatów od różnych producentów może być uciążliwa, a stosowanie osprzętu pomocniczego (styki pomocnicze, wyzwalacze) od innego producenta jest niemożliwe. Za miarę pojemności rozdzielni przystosowanych do montarzu takich aparatów przyjeło się używać liczbę modułów o szerokości 17.5mm (typowy moduł S) Wyłączniki i inne aparaty większych prądów wymagają stosowania innego systemu montarzu.

Można jednak pokusić się o próbę realizacji rozdzielnicy elektrycznej niskiego napięcia i małych prądów (do 125A) w oparciu o (płyką - 350mm głębokości) szafkę rackową. Możliwa jest zatem próba unifikacji budowy punktu dystrybucyjnego w oparciu o jeden typ szaf/stelaży (a w przypadku małych punktów dystrybucyjnych nawet w oparciu o wspólną szafę - wystarczy umieścić tam także panel krosowniczy i switch). Wykorzystać do tego można panele do montażu aparatury modułowej na szynę TS-35 bezpośrednio w szafach rack ("paneli dystrybucji napięć", np. z otworem o szerokości 45.5mm co odpowiada 23 modułom).

Dzięki zastosowaniu dwudzielnej szafy rackowej możliwa jest realizacja rozdzielni elektrycznej składającej się z:

- demontowalnych paneli frotowych o wysokości 3U z wspornikiem DIN do montażu aparatury zabezpieczającej (opisywane powyżej panele dystrybucji napięć)
- wsporników DIN umieszczonych w tylnej części obudowy - umieszczone byłyby tam:

— złącza kablowe (zaciski śrubowe dla przewodów N, L oraz PE, które może być od razu w tym miejscu łączone - także do masy szafy) pomiędzy rozprowadzaną po obiekcie instalacją a dołączonym przewodami typu linka przednim panelem (występują złączki dwu, a nawet trój poziomowe)
— wykonawcze moduły sterujące - przekaźniki, triakowe sterowniki oświetlenia, etc

W szafie tej można (ale przy większych systemach lepiej w osobnej szafie) także umieścić panel RJ-45 na kable sterowania i część decyzyjna sterownika montowane od przodu (w przypadku szaf dzielonych panel RJ45 może być montowany od frontu tylnej części szafy). Optymalnym rozwiązaniem wydaje się zastosowanie szafy dzielonej - zapewniającej swobodny dostęp do złączek po założeniu w przedniej części aparatury modułowej oraz do modułów sterownika, bez jej demontowania. Dla obwodów <= 10A można rozważyć także zastosowanie rozłączalnych kostek łącznikowych (z bolcami - zobacz GP2,5 / WP2,5 w katalog złączek Spółdzielni "Pokój"[1295]), można by je zastosować nawet w formie typowych patchpaneli 1U (podłączenia instalacji) łączonych patchordami z modułami 3+1U (zabezpieczenia wraz z gniazdami podłączeniowymi). Oczywiście takie rozwiązania mają też pewne wady:

— stosowanie złączek kablowych wprowadza dodatkowy punkt łączeniowy, wymaga miejsca, a przy stosowaniu bardziej oszczędnych pod tym względem złączek piętrowych utrudniony jest dostęp do części śrub, można rozważyć zamiast złączek piętrowych zestaw gniazdo przekaźnika + złączki dla N i PE
— stosowanie głębokich dwusekcyjnych szaf typu rack, daje większą powierzchnię instalacyjną, ale tylna powierzchnia mimo wszystko jest trudniej dostępna od frontowej, może być problematyczne lokalizacyjnie (głębokość, wymóg otwierania)

Zobacz też: zdjęcia przykładowej realizacji[1296].

Uwagi ogólne i integracja systemów

Przy tworzeniu kompleksowej instalacji budynku warto zorganizować ją tak aby od centralnego punktu dystrybucyjnego (w którym znajduje się punkt styku z instalacjami zewnętrznymi) nie były prowadzone żadne obce instalacje. Wszystkie media zewnętrzne powinny być wpuszczane w odpowiednie podsystemy tej instalacji wewnętrznej. **Instalacja wewnętrzna** powinna stanowić spójną, dobrze zintegrowaną całość opartą na podobnej architekturze - zarówno na tej samej fizycznej topologii i wspólnych punktach dystrybucyjnych, jak również podobnych mechanizmach działania - wykorzystaniu sieci komputerowych, warto tu rozważyć utworzenie dedykowanych systemów (np. telekomunikacyjny i bezpieczeństwa).

Konieczne jest także zapewnienie odpowiednich warunków dla urządzeń pracujących w punktach dystrybucyjnych (zwłaszcza w głównej serwerowni), a także odpowiedniego bezpieczeństwa tych pomieszczeń. Należy zapewnić także aby opisywane systemy (zwłaszcza system sterowania) były dobrze zintegrowane z **pozostałymi instalacjami technicznymi** (wodno-kanalizacyjną, centralnego ogrzewania, ciepłej wody, klimatyzacjo-wentylacyjną, centralnego odkurzacza, ...).

W tworzeniu zintegrowanego systemu zarządzania obiektem (Building Management Systems[1297]) zachodzi konieczność integracji różnego rodzaju sieci przemysłowych, idealną platformą wydaje się być od strony logicznej warstwa IP, a od elektrycznej Ethernet. Przyjęcie za wspólną płaszczyznę dla wszystkich instalacji nadzorczych i wykonawczych właśnie IP over Ethernet umożliwia łatwą komunikację z dowolnym urządzeniem nie tylko bardziej złożonym sterownikom (potrafiącym programowo obsługiwać poszczególne protokoły), a także tym prostszym poprzez podłączenie stosownych bramek. Ponadto w siec taka można bezpośrednio włączać różnego rodzaju moduły wejść i wyjść cyfrowych lub analogowych - zarówno indywidualne, jak i oparte na wspólnych kasetach.

W tym miejscu chciałbym też uspokoić wszystkich martwiących się o zbyt duże zużycie prądu przez działające non stop systemy automatyki - mocny zestaw komputerowy działający na "jałowym biegu" z wyłączonym monitorem, bez funkcji oszczędzania energii zużywa około 3,36KWh na dobę (przy obecnych cenach mniej niż 1,2 PLN zatem mniej niż gazeta ... oszczędnym czytelnikom pozostawiam wybór); ten sam zestaw w stanie czuwania zużywa około 0,84kWh (mniej niż 30 gr); natomiast switch wraz z systemem sieciowym opartym na mikrokontrolerze ATMega128 wykazał sie tak małym zużyciem prądu że było ono niemierzalne dla używanego do testów licznika.

Instalacja elektryczna

Na instalacje elektryczną składają się:

- rozdzielnice - umieszczane w punktach dystrybucyjnych (tej instalacji), zawierające aparaty zabezpieczające, monitorujące i sterujące
- okablowanie
- osprzęt łączeniowy okablowania - tylko jeżeli decydujemy się na łączenia kabli poza rozdzielnicami
- osprzęt (gniazda, etc) i urządzenia końcowe (odbiory)

Instalacja elektryczna na ogół stanowi hierarchiczny system (kabel zasilający rozdzielnicę jest wyprowadzony w większej rozdzielnicy i tam posiada swoje zabezpieczenia, często także w ramach rozdzielnicy zabezpieczenia odbiorów są jakoś grupowane) dlatego ważne jest zapewnienie odpowiedniego stopniowania i selektywności zabezpieczeń zarówno nadprądowych jak i różnico-prądowych. Nie należy także zapominać o przewidzeniu zastosowania zabezpieczeń przeciwprzepięciowych, systemów sygnalizacyjnych i elementów automatyki - istotne jest zapewnienie styków pomocniczych służących monitorowaniu stanu aparatów, a także zdalnego monitoringu parametrów zasilania (z wykorzystaniem analizatorów z komunikacją modbus lub ethernet) oraz możliwości zdalnego wyłączania czy też włączania (wybranych) zabezpieczeń.

Zasilanie odbiorów końcowych

Rozdzielnie zasilające odbiory końcowe wykonuje się z użyciem tzw. aparatury modułowej. Jej producenci zapewniają szeroki wybór aparatów:

- rozłączniki izolacyjne (FR)
- zabezpieczenia nadprądowe i zwarciowe - m.in.:
  - wyłączniki automatyczne typu S[1298] o różnych charakterystykach (typowo B, C, D, ale nie tylko) i prądach zadziałania
  - wyłączniki silnikowe (z regulowanym prądem zadziałania termicznego członu przeciążeniowego)
  - rozłączniki bezpiecznikowe[1299]
  - podstawy bezpiecznikowe
- zabezpieczenia różnicoprądowe[1300] (RCD, RCCB) wrażliwe na prądy różnego typu: AC - reaguje tylko na przemienne prądy upływu, A - także wyprostowane jednopołówkowe / impulsowe, oraz najbardziej uniwersalny B - reaguje także na prądy stałe
- zabezpieczenia różnicoprądowe z członem nadprądowym (RCBO) - czyli połączenie zabezpieczenia różnicoprądowego z wyłącznikiem typu S
- zabezpieczenia przepięciowe
- aparaty sterujące i sygnalizacyjne takie jak: kontrolki obecności napięcia, wskaźniki, mierniki, przyciski sterujące, przełączniki (FRy?) trzy pozycyjne, styczniki[1301], przekaźniki[1302], …
- osprzęt pomocniczy do zabezpieczeń automatycznych taki jak: wyzwalacze (wzrostowe i zanikowe - powodujące zadziałanie zabezpieczenia odpowiednio przy podaniu na nie napięcia bądź obniżeniu się na ich stykach napięcia poniżej określonego poziomu), styki pomocnicze (służące do sygnalizacji położenia aparatów łączeniowych), napędy do załączania aparatów, …
- elementy automatyki takie jak: przekaźniki impulsowe, sterowniki czasowe, automaty schodowe, zmierzchowe, separatory wejść dla aparatury sterującej (umożliwiające podłączenie wspólnego obwodu głównego do kilku przekaźników), przekaźniki priorytetowe, …

A także osprzętu łączeniowego (szyny grzebieniowe, szpilkowe, bloki łączeniowe, etc.). Niestety systemy aparatury modułowej różnych producentów są ze sobą zazwyczaj niekompatybilne (nawet do tego stopnia że ciężko je połączyć jedną szyną grzebieniową), nie wspominając już o estetyce wyglądu rozdzielnicy, czy też podłączaniu osprzętu pomocniczego - dlatego należy w miarę możliwości starać się wykonywać całą rozdzielnicę aparaturą od jednego producenta.

Wspomniane wyżej, dostępne na rynku elementy automatyki do rozdzielnic elektrycznych są najczęściej rozwiązaniami wysokonapięciowe i nie umożliwiającymi bardziej zaawansowanego zdalnego sterowania. Dlatego uważam iż bardziej zaawansowane systemy automatyki instalacji elektrycznej należy opierać na dedykowanych sterownikach mikroprocesorowych, w których sterowanie odbywa się niskonapięciowo, a ich funkcja jest zmieniana programowo. Natomiast zastosowanie opisywanej aparatury modułowej systemów sterowania powinno sprowadzać się głównie do funkcji pomocniczych wobec podstawowych aparatów łączeniowych - umożliwienia uzyskanie informacji o ich stanie oraz powodowania zadziałania, a także wizualną sygnalizację parametrów pracy rozdzielnicy (lampki, wskaźniki i liczniki). Na stronie opisuję także wielokanałowy kanałowy mikroprocesorowy sterownik oświetlenia z ściemniaczem.

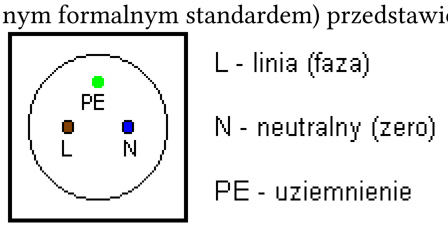
Istnieją dwa podstawowe sposoby prowadzenia instalacji elektrycznych:

— topologia magistralowa - poszczególne odbiorniki podłączane są do wspólnego kabla wychodzącego z rozdzielni i posiadają tam wspólne zabezpieczenie
— topologia gwiaździsta - każdy odbiornik zasilany jest osobnym przewodem prowadzonym z rozdzielni

Większość instalacji stanowi swego rodzaju miks tych dwóch podejść - z rozdzielni wychodzi (gwiaździście) kilka obwodów magistralowych. Ze względów na możliwości sterowania, czytelnośc instalacji, itd. lepszym rozwiązaniem wydaje się topologia gwiaździsta. Dlatego instalację należy starać się prowadzić w sposób gwiaździsty od rozdzielnicy (punktu dystrybucyjnego), ale względu na ilość dość grubych przewodów należy starać się robić niezbyt wielkie rozdzielnice w pobliżu odbiorów (łatwiej poradzić sobie z jednym przewodem o większym przekroju do zasilania rozdzielnicy niż wychodząca z niej wiązką przewodów). Należy także pamiętać o zachowaniu odpowiednich odległości między wysoko-prądowymi instalacjami zasilającymi a telekomunikacyjnymi. Konieczne jest także zapewnienie odpowiedniego rozdziału obciążenia między fazy.

Warto dla każdej linii stosować indywidualne zintegrowane zabezpieczenie różnico i nad-prądowe (RCBO), a obwody grupować dopiero na następnym (z zachowaniem selektywności) stopniu zabezpieczeń. W przypadku podłączania pod jeden RCD kilku zabezpieczeń nadprądowych warto rozważyć stosowanie zabezpieczeń dwutorowym (jest to ważne przy usuwaniu niektórych z uszkodzeń powodujących działanie RCD - konkretnie upływności N-PE, przy których załączenie dowolnego z obciążonych S-ów za RCD powoduje zadziałanie RCD) - dostępne są także w wykonaniu 1 modułowym (zobacz w Sieci: Ciekawe rozwiązania aparatów modułowych[1303]).

Warto stosować powszechnie uznawany układ faza/neutralny w gniazdkach 230V (ale nie będący chyba żadnym formalnym standardem) przedstawiony na ilustracji:



gniazo z uziemnieniem

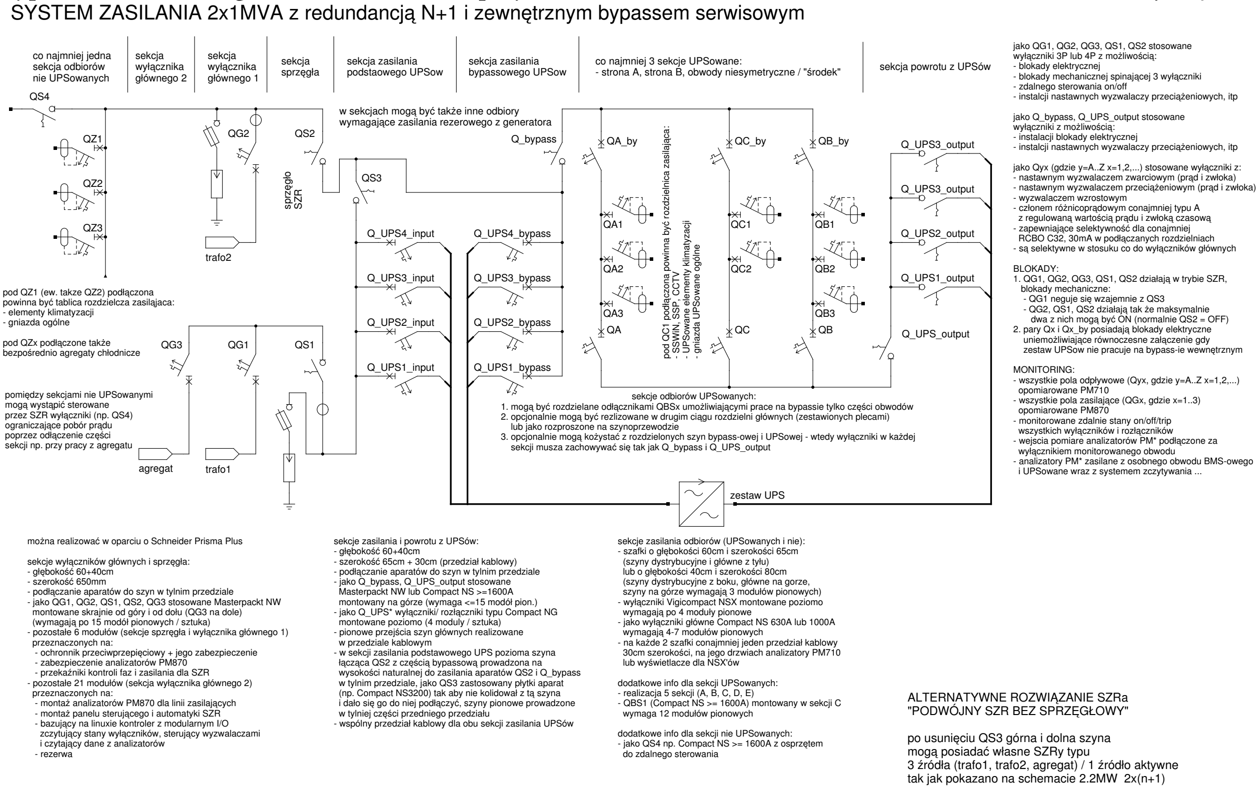
Główny system zasilania obiektu

Przy projektowaniu głównego systemu zasilającego w obiekcie (rozdzielni głównej) bardzo istotne jest zapewnienie odpowiedniego podziału systemu na pod-rozdzielnie z uwzględnieniem ich ewentualnego zasilania zapasowego (UPS, agregat), redundancji zasilania, rozkładu obciążenia, możliwości rozbudowy i modernizacji (stosowny zapas miejsca w rozdzielni, modułowość jej wykonania). Zamieszczam kilka koncepcji rozdzielni głównej dla obiektu infrastruktury komputerowej prezentujący kilka wartych uwagi rozwiązań.

redundancja N+1

Układ typu N+1 z niezależnymi ścieżkami dystrybucji posiada dwa tory wejściowe po 1MW z układem SZR (z obsługą agregatu), redundancję N+1 na UPSach i trzy tory zasilania serwerów (A i B dla odbiorników z dwoma zasilaczami oraz C dla odbiorników z jednym zasilaczem), z których każda może być niezależnie przełączona na

bypass. Dla układu tego zamieszczam także przykład rozmieszczenia[1304] rozdzielni wraz z UPSami i klimatyzacją.

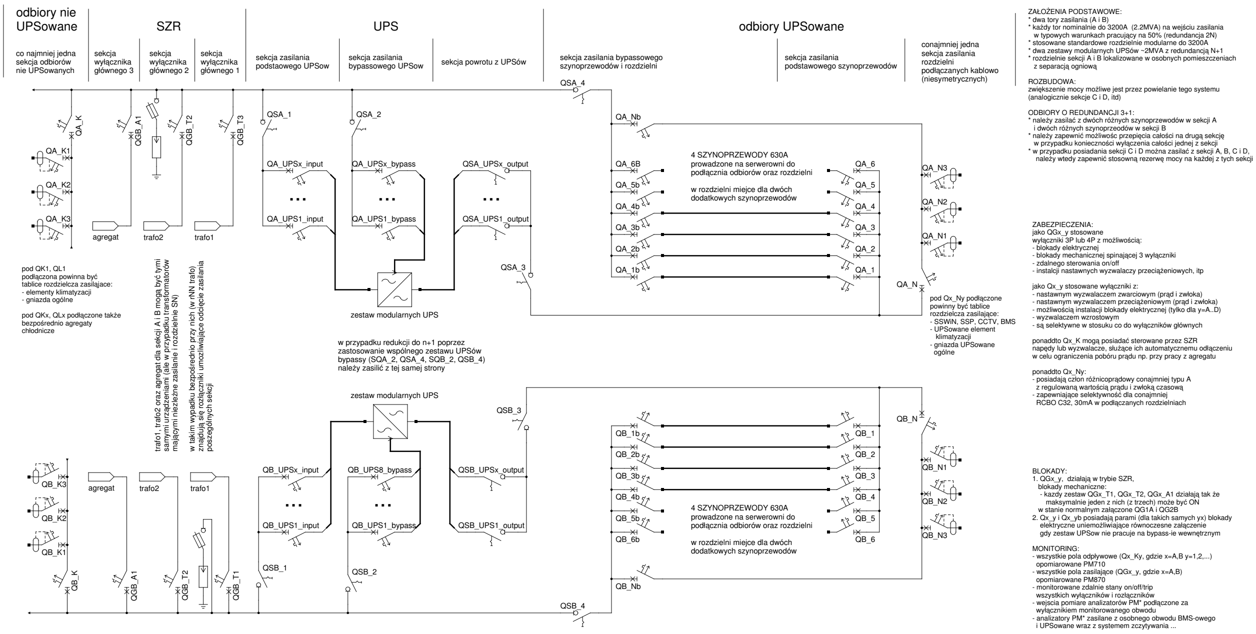


REDUNDANCJA 2(N+1)

Układ typu 2(N+1) z niezależnymi ścieżkami dystrybucji posiada dwa tory wejściowe po 1MW (plus agregat) z innym rozwiązaniem układu SZR, zasilenia UPSów i dystrybucji prądu UPSowanego. Każda strona zasilania (A i B) jest całkowicie niezależna od drugiej (w szczególności posiada własną rozdzielnie główną, układ SZR, system UPSów oraz niezależne tory dystrybucji aż do redundantnego zestawu zasilaczy w odbiorniku), aby to zapewnić pomieszczenia rozdzielni, UPSów, akumulatorów poszczególnych stron muszą być także całkowicie niezależne (izolowane od siebie strefy ogniowe). Optymalnym (m.in. ze względu na standardowe wykonania rozdzielni oraz aparatów) wydaje się realizowanie zasilania jako zestawu systemów 2(n+1) o mocy nominalnej rzędu 2.2MW (3200A). Dystrybucję zasilania po serwerowni realizowana jest z wykorzystaniem szynoprzewodów 630A i podłączanych do nich rozdzielni dedykowanych pod dany system komputerowy. Oba końce szynoprzewodu powinny być podłączane do rozdzielni głównej - jeden do części UPSowanej, drugi do części bypassowej. Obwody gniazd ogólnych realizowane są w postaci indywidualnie zabezpieczanych gniazd trójfazowych 32A i jednofazowych

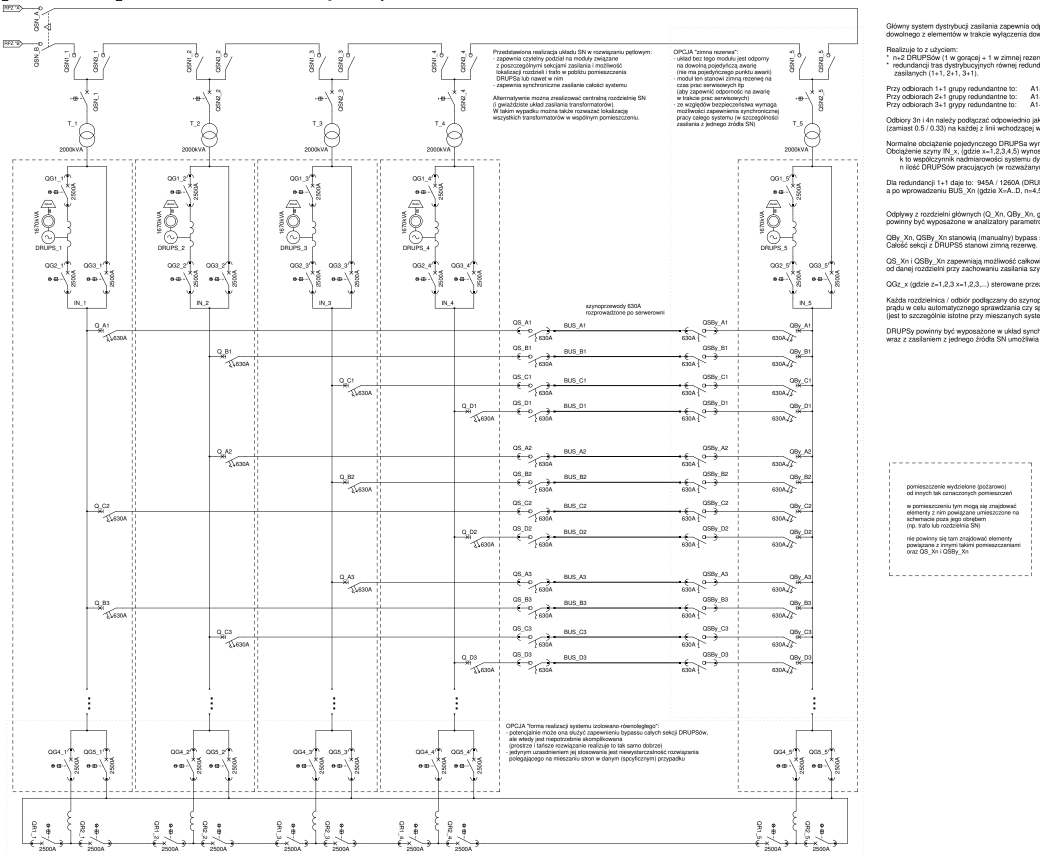
16A zgodnych z standardem IEC 60309[1305], rozdzielnie je zasilające są podłączone bezpośrednio do rozdzielni głównej.

SYSTEM ZASILANIA 2.2MVA z redundancją 2(N+1)



REDUNDANCJA N+2

Układ typu N+2 jest alternatywne rozwiązanie zapewniającym (przy poprawnym wykonaniu) nie mniejszy poziom odporności na awarie jak system 2(n+1).



Jak już wspomniano w niektórych zastosowaniach kluczowe przy projektowaniu systemu zasilania jest zapewnienie jego dużej niezawodności oraz redundancji (zarówno celem zwiększenia odporności na awarie, jak i celem umożliwienia serwisowania systemu w trakcie pracy). Możemy to uzyskać zwielokrotniając tory zasilania (np. redundancja 2N gdzie mamy 2 identyczne tory zasilania, z których każdy jest w stanie przyjąć pełne obciążenie) lub elementy w torach zasilania (np. N+1, gdzie potrzebujemy N jednostek - np. modułów systemu UPS a mamy jedną

nadmiarową). Ciekawym rozwiązaniem jest redundancja N+1 lub N+2 oparta na całych torach zasilania (oczywiście aby było to bardziej atrakcyjne od 2N N musi być większe od 1, ale w dużych systemach na ogół nie jest to problem).

Urządzenie o mocy $n * Y$ potrzebuje:

- Y podstawowych linii zasilających o mocy n każda
- X redundancyjnych linii zasilających o mocy n każda

Typowo (Y, X) = (1, 0), (1, 1), (1, 2), (2, 1), (2, 2) lub (3, 1). Najczęściej spotykane są urządzenia typu (1, 1) i (1, 0). Zasilanie w układzie (3,1) jest typowe dla dużych jednostek obliczeniowych. Przypadek (1, 1) i (2, 2) oznacza redundancje typu "2N", przypadek (1,2) oznacza redundancje typu "3N". Przypadek (2, 2) jest równoważny przypadkowi (1, 1) gdyż i tak nie mamy odporności na zanik dowolnych dwóch linii.

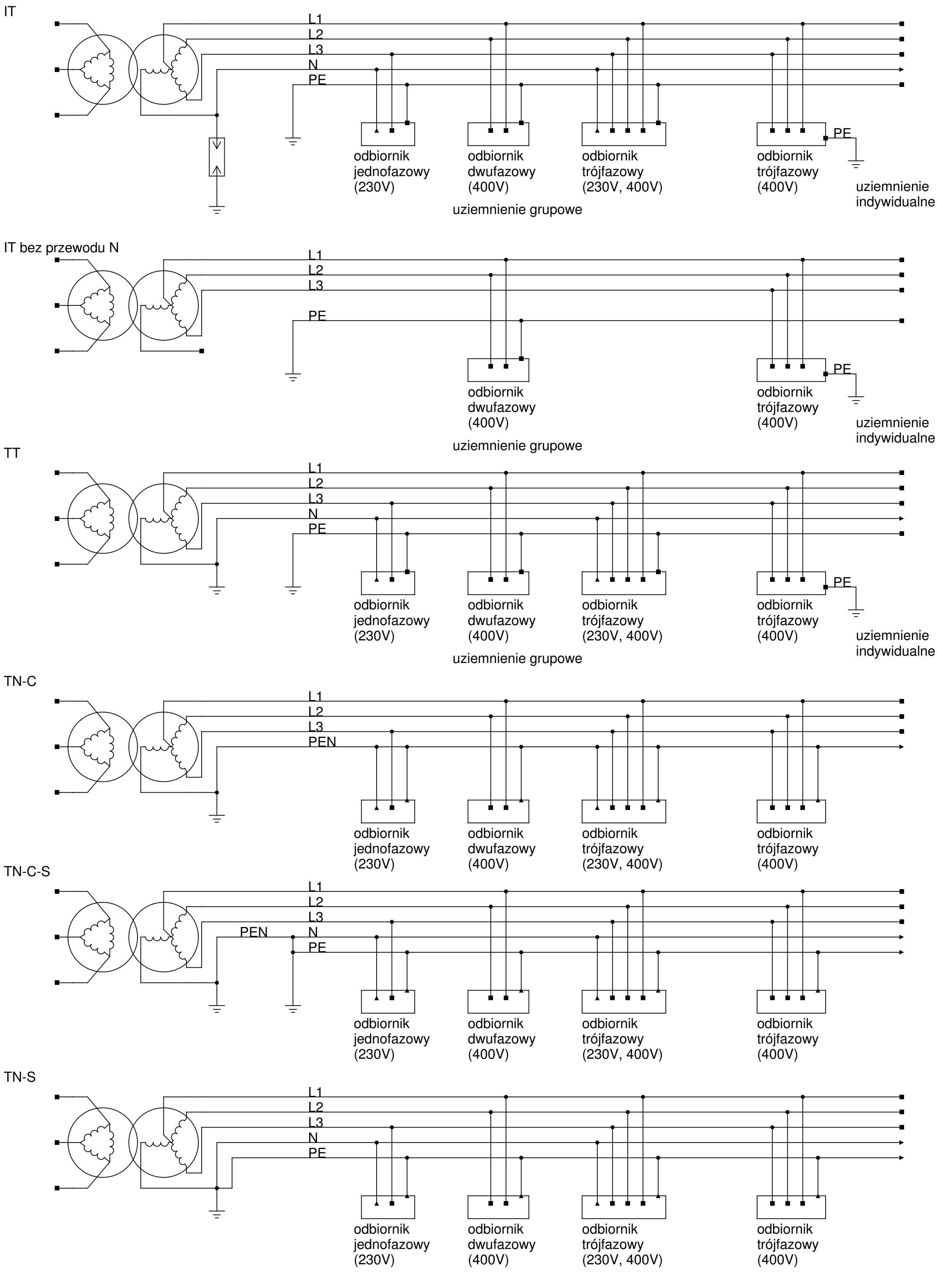
Liczbę potrzebnych linii zasilających determinuje $max((Y + X)/nwd(x, y))$. Czyli dla wszystkich typowych przypadków wystarczający jest 4 liniowy system zasilania (strony: A, B, C, D).

Do systemu musimy dostarczać $N = sum(n * Y)$ mocy z zachowaniem wymaganej redundancji. Zatem potrzebujemy $N/M + max(X)$ układów zasilających o mocy M każdy. Jeżeli urządzeń większym X od reszty jest niewiele to rezerwę redundancyjną dla nich możemy ukryć w N/M, czyli potrzebujemy: $(N + N')/M + X'$, N' to odpowiednia część mocy urządzeń z wymaganą redundancją większą niż X'.

Ponadto (zwłaszcza gdy $X' = 1$ lub wiele urządzeń ma $X = 1$) warto zastosować jeszcze jeden dodatkowy system zasilania w trybie tak zwanej zimnej rezerwy, który zapewni zachowanie redundancji w trakcie prac serwisowych na dowolnym z układów podstawowych. W ten sposób otrzymujemy redundancje systemów zasilania: $(N + N')/M + X' + 1$.

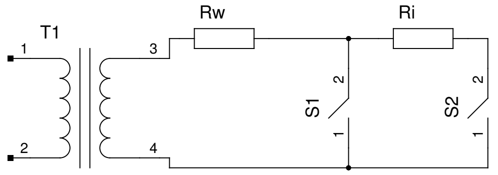
Układy sieci - IT, TT, TN

Wyróżnia się 3 podstawowe układy sieci zasilającej[1306]: IT (punkt neutralny transformatora izlowoany, uziemnienia urządzeń łączone do ziemi), TT (punkt neutralny transformatora połączony z ziemią, uziemnienia urządzeń łączone do ziemi), TN (punkt neutralny transformatora połączony z ziemią, uziemnienia urządzeń łączone do przewodu neutralnego i uziemniane wraz z nim). W tym ostatnim wyróżnia się jego odmiany związne z rozdzieleniem przewodu PE (uziemnienie ochronne) i N (roboczy przewód neutralny) od przewodu wspólnego PEN.



Prąd zwarcia

Przy projektowaniu instalacji i doborze zabezpieczeń istotna wrtość odgrywa prąd zwarcia. Oblicza się go w oparciu o dane transformatora i parametry linii przesyłowych (grubość i materiał kabli, co przekłada się na ich rezystancję) zgodnie z poniższymi zasadami:



parametry transformatora:
* prąd nominalny = In =prąd uzwojenia wtórego przy nominalnym obciążeniu transformatora
* napięcie zwarcia = Uk = napiecie na uzwojeniu pierwotnym powodujace przeplyw nominalnego pradu na uzwojeniu wtorym
  Uk[%] wyrażone jako procent nominalnego napięcia uzwojenia pierwotnego Uk[%]=Up/Uk
* rezystancja wewnętrzna = Rw

prąd zwarcia w obwodzie zamykanym przez S1 (Ik1) obliczamy w oparciu o parametry T1:
 Ik1 = In / Uk[%]

w oparciu o napięcie nominalne strony wtórnej Un możemy obliczyć Rw
  Rw = Un / Ik1 = Un*Uk[%] / In

dzięki temu możemy obiczyć prąd zwarcia instalacji:
  Ik2 = Un / (Rw + Ri) = Un / (Un*Uk[%]/In + Ri)

przyklad:
1. trafo na 400V o mocy 1.2MW (In=1800A) i napięciu zwarcia Uk[%]=6%
2. szyna Cu o przekroju 800mm2 i dlugosci 1m (R = 0.00004545Ohm)
3. zatem Ik2 = 400 / (400*0.06/1800 + 0.00004545) = 29898A

Obwody prądu zmiennego

Szeregowy obwód RLC}

Prąd:
$I_{sk} = U_{sk} / \sqrt{R^2 + \left(\omega L - \frac{1}{\omega C}\right)^2}$, gdy brak cewki $L = 0$, brak kondensatora $C = \infty$
Kąt przesunięcia fazowego:
$\phi = \arctan\left(\frac{X_L + X_C}{R}\right) = \arctan\left(\frac{\omega L - \frac{1}{\omega C}}{R}\right)$
Moc wydzielana:
$P = U_{sk} I_{sk} \cos(\phi)$

alternatywnie w oparciu na zależność związaną z czasem ładowania kondensatora stałym prądem można przybliżyć tą wartość przez:
$P = R\left(\frac{C\Delta U}{\Delta T}\right)$
Skrypt dla calc:

```
U=230
OM=2*3.14/50
L=0
C=100e-09
R=1000
X=OM*L - 1/(OM*C)
I=U*(R^2+X^2)^-0.5
cosFI=cos(atan(X/R))
U*I*cosFI
```

Moc pozorna, czynna i bierna

Moc pozorna:
$S = U_{sk} I_{sk} = U_m I_m / 2$
$S = Z I_{sk}^2$
$S = \sqrt{P^2 + Q^2}$
Moc czynna:$P = U_{sk} I_{sk} \cos(\phi)$
Moc bierna:$Q = U_{sk} I_{sk} \sin(\phi)$
Zobacz także: skrypt gnuplota ilustrujący zależności między mocą pozorna, czynna i bierna

Reaktancja

$Z = R + jX$ - odpowiednik oporu dla elementów z urojoną impedancją:

- kondensator: $X = -\frac{1}{\omega C}$
- indukcyjność: $X = \omega L$

Oznaczenia

- U_{sk}, I_{sk} - wartości skuteczne napięcia i prądu
- U_m, I_m - amplitudy napięcia i prądu

Zabezpieczenia przepięciowe

Interesującym i ważnym zagadnieniem z pogranicza elektroniki i elektrotechniki są zabezpieczenia przepięciowe.

W układach elektronicznych najczęściej stosowane są diody zabezpieczające (określane mianem transil[1307]). Ich działanie jest podobne do diody Zenera - montowane są zaporowo i po wzroście napięcia do wartości znamionowej gwałtownie przewodzą, pozwalają przy tym na wytracenie znacznej mocy przez krótki czas. W tym czasie prąd płynący przez transil powinien wywołać zadziałanie bezpiecznika i odcięcie urządzenia od zasilania. Dostępne są zarówno transile jedno jak i dwu kierunkowe. Innym spotykanym zabezpieczeniem jest warystor[1308] będący rezystorem o wartości oporu silnie zależnej od przyłożonego napięcia. Jest on elementem podobnym w działaniu do transila tyle że zawsze dwukierunkowy i wolniejszy.

W zabezpieczeniach instalacji elektrycznej (ochronniki przepięciowe[1309]) wyróżnia się 4 stopnie ochrony A (wymagany tylko przy liniach napowietrznych), B (w tablicy głównej budynku), C (w tablicy mieszkania, pomieszczenia), D (do ochrony urządzeń wrażliwych - np. elektroniki, stosowany w listwach przepięciowych[1310]). Jako zabezpieczenia stouje się warystory metal-oxygen (końcowe stopnie ochrony) oraz iskrowniki. Istotną rolę odgrywa również instalacja piorunochronowa[1311] działająca na kształt (mocno nieidealnej) klatki Faradaya[1312].

Dość zaskakujące wydaje się stosowanie w układzie sieciowym[1313] TN-S zabezpieczeń o dużej wartości znaminowej napięcia pomiędzy N a PE, których potencjał powinien być bliski. W zasadzie można by sobie wyobrazić konstrukcję zabezpieczenia N-PE o znacznie mniejszym napięciu znamionowym - np. jako warystor (np. 14V) lub szeregowo połączone transil dwukierunkowy (np. 10V, 5W) i rezystor (np. 15Ohm, 5W); w takim przypadku należy tez rozważyć układ wyłączający całość gdy przez takie zabezpieczenie płynie prąd (coś na kształt RCD).

Linki

Zobacz w Wikipedii: Instalacja elektryczna[1314], Rozdzielnica elektryczna[1315], Bezpiecznik elektryczny[1316], Bezpiecznik różnicowo-prądowy[1317], Gniazdko elektryczne[1318], Łączniki elektryczne[1319], Electrical room[1320], Electric switchboard[1321], Utility tunnel[1322], AWG[1323], Industrial lamps[1324]. Zobacz w Sieci: Publikacje SEP[1325], Instalacje elektryczne na terenach wiejskich[1326].

Galeria zdjęć związanych z instalacją elektryczną[1327]

System automatyki i sterowania

Istnieje kilka "profesjonalnych" systemów sterowania tzw. "inteligentnym budynkiem[1328]" - takich jak KNX[1329] (dawniej EIB), LonWorks, BACnet[1330] (trochę więcej na temat poszczególnych standardów w interfejsy elektroniki). Jednak w chwili obecnej nie można powiedzieć że któryś z nich jest "prawdziwym" standardem (takim jak np. w sieciach komputerowych jest ethernet); sytuacja taka z jednej strony pozostawia sporą dowolność przy projektowaniu "swojego" systemu, z drugiej wymaga zwrócenia uwagi na możliwość współpracy "naszego" systemu z różnorakimi standardami - w szczególności wspomnianym już Ethernetem (TCP/IP) oraz popularnym standardem protokołu komunikacyjnego typu RS-* (komputerowy RS-232[1331] lub przemysłowe RS-485 i podobne). Warto zwrócić także uwagę na współpracę z wartymi uwagi otwartymi protokołami komunikacji sterowników takimi jak wspomniany BACnet czy KNX. W bardziej rozbudowanych systemach zachodzi często potrzeba zapewnienia współpracy

pomiędzy różnymi standardami sieci przemysłowych - idealną platformą wydaje się tutaj Ethernet wraz z protokołem IP, który został stworzony do łączenia różnych sieci.

Osobiście mam dość sceptyczne podejście do systemów komunikacji wykorzystujących kable energetyczne i nie przemawia do mnie argument że ułatwiają one montaż automatyki w istniejącej instalacji - jestem zwolennikiem rozwiązań kompleksowych (pełnej modernizacji instalacji elektrycznej oraz instalacji teleinformatycznych i teletechnicznych). Preferuję gwieździste prowadzenie dedykowanych linii punkt-punkt pomiędzy każdym ze sterowników a punktem dystrybucyjnym (także w stosunku co do interfejsów tego nie wymagających np. RS-485, gdyż takie rozwiązanie przy rozsądnym doborze przewodu umożliwia łatwa zmianę standardu komunikacji z danym sterownikiem), a nawet stosowanie gwiazdy wobec instalacji elektrycznej co umożliwia montowanie większości aparatury systemów automatyki w punkcie dystrybucyjnym. Bardzo dobrym i elastycznym rozwiązaniem okablowania sterującego jest Ethernet (wraz z TCP/IP), jest to rozwiązanie bardzo elastyczne i wszechstronne (od komunikacji z sterownikiem oświetlenia do przesyłu obrazów z kamer przemysłowych) niestety również w chwili obecnej dość kosztowne (zwłaszcza przy aplikowaniu go w układy własnej produkcji).

Układ taki (gwiazda) skłądnia do stosowania wyróżnionego centralnego mastera lub centralnego mastera i zespołu "pod masterów" umieszczanych w niżej położonych w hierarchii węzłach drzewa. Funkcję centralnego mastera powinien pełnić komputer typu PC najlepiej z jakimś linuxem/*bsd (stabilność takich systemów jest na prawdę na tyle duża że nie należy zbytnio martwić się o zawieszenia, ale zawsze można zdublować mastera - mogłyby się nawet wzajemnie pilnować), daje to bardzo dużą elastyczność - system cechuje się dużymi zdolnościami rekonfiguracyjnymi, możemy łatwo korzystać z zdalnego sterowania systemem przez Internet (np. przez protokół HTTPs[1332] wraz z autoryzacją HTTP lub SSH[1333]). Rolę "pod masterów" mogą pełnić systemy oparte na mikrokontrolerach np. AVR lub ARM (najlepiej z interfejsem ethernetowym), również w oparciu o mikrokontrolery realizowane mogą być układy wykonawcze/sterujące (im do komunikacji z "submasterem" wystarczy interfejs typu RS-485). Rozwiązanie takie jest analogiczne do stosowanych w sieciach IP (które wykazywały się już swoją elastycznością i odpornością w różnych sytuacjach) gdzie każdy host może komunikować się niezależnie z innym, ale są urządzenia i hosty pełniące specjalną funkcję - switche / routery umożliwiające komunikację jakiejś podgrupie hostów oraz serwery usług (np. DNS), a bezpieczeństwo na wypadek awarii zapewnia się poprzez dublowanie tych systemów (zapasowy serwer DNS czy też awaryjne łącze zewnętrzne). Układy wykonawcze powinny mieć także lokalne obwody sterujące (przyciskowe), oczywiście z zachowaniem możliwości współpracy poprzez mastera z oddalonymi układami sterowania (zespołami przyciskowymi czy też panelami operatorskimi).

Okablowanie od urządzeń automatyki i bezpieczeństwa powinno być terminowane w punkcie dystrybucyjnym gniazdami RJ-45. Każde gniazdo na panelu krosowym odpowiada pojedyńczemu urządzeniu (max dwóm identycznym urządzeniom) i zawiera wszystkie sygnały danego urządzenia (w tym oba bieguny zasilania).

Zobacz także projekty związane z automatyką domową, monitoringiem systemów.

Zobacz w Sieci: Inteligentny dom zarządzany przez sterownik PLC[1334], zegaruz[1335], DomNet[1336], EIB[1337], Serwery portów szeregowych[1338], Ethernet przemysłowy[1339], 110 block[1340], Keystone module[1341].

Systemy bezpieczeństwa

Do kategorii systemów bezpieczeństwa zalicza się systemy:

- sygnalizacji pożaru (SSP), a także związane z nimi automatyczne systemy gaszące
- sygnalizacji włamania i napadu (SSWiN)
- kontroli dostępu (KD)
- telewizji dozorowej (CCTV)

Przy planowaniu tego typu systemów kluczową kwestią jest rozmieszczenie czujników, kamer, klawiatur sterujących itp. Bardzo istotną kwestią jest także zapewnienie współpracy tych systemów ze sobą i pozostałymi systemami obiektu oraz zapewnienie możliwości ich monitoringu (a także w pewnym stopniu odpowiednio zabezpieczonego sterowania) z użyciem sieci teleinformatycznej IP i ustandaryzowanych, otwartych protokołów.

Najprościej jest w przypadku **systemu kontroli dostępu** - punkty rozmieszczenia czytników, klawiatur, elektrozaczepów do zamków czy też zwór magnetycznych określa położenie drzwi prowadzących do poszczególnych stref i wyznaczenie samych tych stref.

W przypadku **systemów alarmowych**[1342] sytuacja jest nieco trudniejsza, ale nie jest to jeszcze tak wielkim problemem. Czujki systemu antywłamaniowego takie jak czujniki ruchu (PIR - reagujące na zmiany promieniowania podczerwonego w obserwowanym obszarze (ruch obiektów emitujących ciepło), mikrofalowe - działające na zasadzie efektu doplera, itp.) umieszcza się tak aby monitorowały w miarę możliwości całe pomieszczenie, a przynajmniej wejścia (nie należy zapominać o oknach itp). Warto system wzbogacić o możliwość kontrolowania otwarcia okien i drzwi (kontaktrony, zmieniające położenie styków w przypadku otwarcia drzwi), a także bardziej specjalistyczne zabezpieczenia jak bariery podczerwieni (reagujące na przerwanie wiązki), czujniki akustyczne (np. reagujące na

dźwięki tłuczonego szkła), nacisku podłoża czy też drgań bądź zmiany ciśnienia w pomieszczeniu (związanej z otwarciem dostępu do niego). Detektory systemu przeciwpożarowego, takie jak czujki dymu (zarówno standardowe czujki optyczne, jak i czujki zasysające przystosowane do pracy w pomieszczeniach z systemami wentylacyjnymi, a także posiadające większą czułość od konwencjonalnych), czujki termiczne, itp montuje się w taki sposób aby pokrywały odpowiednio gęsto cały obiekt, z kolei w miejscach narażonych na występowanie niebezpiecznych gazów (metan, butan, propan, tlenek węgla itp) stosuje się stosowne detektory gazów. W obu wypadkach stosowane są także ręczne powiadamiacze - napadów lub pożarowe i odpowiednie urządzenia sygnalizacyjne (akustyczne i świetlne). System przeciwwłamaniowy zawsze posiada także manipulatory (klawiatury) służące do jego uzbrajania i rozbrajania.

Wśród czujek można wyróżnić dwa podstawowe typy pod względem komunikacji z centralą - czujniki nieadresowalne (czujnik przełącza zestyki w przypadku detekcji) oraz czujniki adresowalne (czujnik komunikuje się cyfrowo z centralą identyfikując się i zgłaszając alarm). Zaletą drugiego rozwiązania jest łatwa identyfikacja z której czujki pochodzi alarm, wadą niekompatybilność protokołów komunikacyjnych i często objęcie ich tajemnicą, z kolei w pierwszym przypadku problem ten nie występuje, ale identyfikowalność czujki wymaga rozbudowanego okablowania (gwiazda, którą i tak polecam …) i centrali z wieloma wejściami.

Od strony logicznej racjonalne mogą wydawać się dwa podejścia - rozproszenie inteligencji w urządzeniach do takiego stopnia aby czujniki mogły bezpośrednio komunikować się po szyfrowanym IP z centralą (gwiaździsta struktura), albo stosowanie zwykłych (nie inteligentnych czujników) i podcentralek (obsługujących dane pomieszczenie) zapewniających szyfrowaną komunikację IP z centralą (struktura drzewa).

Najbardziej skomplikowana sytuacja ma miejsce w przypadku systemów **telewizji przemysłowej** (CCTV[1343]) - nie ma praktycznej możliwości pokrycia tak aby móc nieprzerwanie rejestrować z dobrą jakość każdy fragment obiektu, dlatego też konieczne jest określenie rejestracją czego (jakich miejsc) jesteśmy zainteresowani. Kolejnymi pytaniami na które należy udzielić odpowiedzi przed zaprojektowaniem systemu CCTV jest czy system ma być sterowany (możliwość zdalnego poruszania kamerą oraz robienia zbliżeń) oraz czy chcemy rejestrować w sposób ciągły czy tylko gdy wystąpiło zdarzenie (detekcja ruchu na kamerze lub zadziałanie czujnika). System sterowany ma większe możliwości przy śledzeniu obiektów, często jest bardziej funkcjonalny przy prowadzeniu monitoringu "na żywo", natomiast gorzej wypada w rejestracji zwłaszcza samodzielnej - bez udziału operatora (z kamery obrotowej uzyskamy mniej informacji niż z dwóch dobrze rozmieszczonych kamer stacjonarnych). Dlatego też warto dobrze zastanowić się czy naprawdę potrzebujemy systemu sterowanego (koszt aparatury sterowniczej do jednej kamery odpowiada kosztowi 1 do 3 dodatkowych kamer), oczywiście są sytuacje gdzie system sterowany jest w zasadzie niezbędny - np. policyjny monitoring miast. Można także (co często jest najlepszym wyjściem) zdecydować się np. na system nie sterowalny wzbogacony kilkoma kamerami sterowalnymi (tam gdzie jest to niezbędne). W miejscach gdzie jakość (zwłaszcza kolorowego - wierność oddania kolorów) nie jest sprawą kluczową można stosować kamerki CMOS, w pozostałych (zwłaszcza tam gdzie zależy nam na dobrej pracy kamer noc/dzień i wiernej kolorystyce) polecam stosować kamery CCD. Szczególną uwagę warto zwrócić na wygodne w stosowaniu zintegrowane kamery noc/dzień zawierające w jednej hermetycznej obudowie także oświetlacz podczerwieni, a niekiedy także obiektyw z regulowaną ogniskową.

Współcześnie dzięki powszechności kamer typu noc/dzień (które przy dobrym oświetleniu działają jak kolorowe, przy słabszym działają jak czarno-białe i mogą rejestrować w podczerwieni) nie wystepuje już problem wyboru pomiędzy systemem kolorowym a czarnobiałym. Innym zagadnieniem jest rejestracja - w dzisiejszych czasach w praktyce już wyłącznie cyfrowa, natomiast problem wiąże się z miejscem digitalizacji: w kamerze (sieciowe kamery IP, rozwiązanie kosztowne, ale widać znaczący postęp w dostępności profesjonalnych kamer IP), w centrali monitoringu (rozwiązanie kłopotliwe przy większych systemach ze względu na okablowanie), w rozproszonym systemie rejestratorów / serwerów wideo umieszczonych w poszczególnych punktach dystrybucyjnych połączonych dedykowaną siecią komputerową (w chwili obecnej nadal chyba najlepsze rozwiązanie).

Zagadnienie punktu w którym dokonujemy digitalizacji obrazu z CCTV (kamery IP czy serwer z kartą framegrabbera/DVR) można porównać do sytuacji w systemach alarmowych - można by inwestować w budowę drogich czujek komunikujących się cyfrowo (np. po Ethernet+IP lub po RS485), ale bardziej racjonalne wydaje się zastosowanie prostych czujek przesyłających analogowo podstawową informację (ew. cyfrowych termometrów 1wire, gdyż ich koszt nie jest porażający) oraz zaawansowaną centralę której oprócz transmisji sygnałów w sposób cyfrowy (IP over Ethernet, RS485) możemy powierzyć inne zadania. Dość interesującym pomysłem wydaje się być zbudowanie takiej centralki w oparciu o dedykowaną kartę / interfejs szeregowy z optoizolowanymi wejściami, na bazie serwera przechwytującego wideo. Kolejną związaną z rejestracją kwestią jest czy opierać ją na autonomicznych rejestratorach/serwerach wideo, czy wykorzystać do tego celu platformę PC z specjalnymi kartami - pierwsze rozwiązanie wymaga mniejszych nakładów konfiguracyjnych, drugie - (porównywalne cenowo, niekiedy tańsze) jest bardziej elastyczne (jak wspomniałem ten sam komputer może nadzorować daną strefę systemu alarmowego, można bez większych trudności dublować systemy czy wprowadzać systemy zapasowe). Oczywiście w pewnych sytuacjach warto się pokusić o digitalizację bezpośrednio w kamerze, a także bezpośrednio w czujce (adresowalnej, najlepiej komunikującej się w jakimś powszechnie znanym protokole - jak RS485 czy Ethernet). Rozwiązanie takie ma sens głównie gdy chcemy podnieść bezpieczeństwo poprzez zastosowanie szyfrowania i cyfrowego podpisywania przesyłanych komunikatów, na przykład z wykorzystaniem mechanizmów PGP.

Zaletami cyfrowych systemów monitoringu (zwłaszcza z wczesną digitalizacją) obok łatwiejszej integracji z innymi systemami, łatwiejszej realizacji technik inteligentnych jest możliwość szyfrowania transmisji i stosowania kamer megapixelowe (ich rozdzielczość jest przydatna przy identyfikacji osób, istotne jest także większe pole obserwacji z jednej kamery, co umożliwia ograniczenie liczby kamer w systemie). Systemy takie umożliwiają także niezależność obserwacji od rejestracji (w tym ukrywania przed operatorem fragmentów/kamer przy jednoczesnej rejestracji tych obrazów), ponadto pozwalają na zastosowanie technik cyfrowego przetwarzania obrazów - wspomnianych już technik inteligentnego monitoringu, ale także filtracji celem poprawy jakości obrazu lub uwydatnienia szczegółów, kompresji umożliwiającej wydajniejszą transmisję i składowanie.

Dobra kamera IP powinna charakteryzować się:

— przetwornikiem CCD
— trybem noc/dzień
— rejestracją i transmisją obrazu co najmniej w rozdzielczości 704*576 (PAL, 4CIF) przy 25kl/s (zalecana rozdzielczość megapixelowa)
— rejestracją i transmisją audio
— strumieniowaniem obrazu i dźwięku poprzez RTSP[1344]
— udostępnianiem obrazu JPEG i MJPEG poprzez HTTP[1345]
— wbudowanym lub dołączanym oświetlaczem IR
— możliwością stosowania różnych obiektywów, najlepiej jakby standardowo była wyposażona w obiektyw "auto iris" ze zmienną ogniskową (conajmniej 2.8-12mm, co odpowiada kątowi widzenia 24-96 stopni)
— niezależnym od przeglądarki zarządzaniem poprzez HTTP (przydatne byłoby także zarządzanie po SSH lub SNMP)
— możliwością realizacji detekcji ruchu w kamerze wraz z wysyłaniem związanych z tym sekwencji na FTP lub email oraz zapisem wideo na serwerze NFS lub FTP z uwzględnieniem pre-bufora (x sekund obrazu z przed detekcji ruchu)
— wyjściem do sterowania obrotnicy i obiektywu z zoomem kontrolowanym poprzez sieć
— wejściami i wyjściami alarmowymi z możliwością sterowania zdalnego oraz lokalnego (system detekcji ruchu w kamerze))
— kanałem zwrotnym audio (do kamery) realizowanym poprzez SIP

Warte zainteresowania wydają się także **"inteligentne systemy monitoringu"** w których to oprogramowanie jest odpowiedzialne za automatyczną detekcję zagrożeń takich jak naruszenie stref bezpieczeństwa (wykrywanie w nich ruchu lub obecności), wykrywanie zostawione bądź zabranego przedmiotu. Systemy takie mogą posłużyć także do śledzenia (obserwacji trajektorii) i identyfikacji obiektów oraz do wykrywania zachowań podejrzanych (panika, bójka, gromadzenie się tłumu, …) czy też przesunięcia kamery. Głównym problemem w takich systemach jest dobra eliminacja tła oraz wydzielenie obiektów z obrazu.

Również nie małe pole zastosowań w systemach nadzoru i kontroli dostępu znajdują **techniki biometryczne** takie jak identyfikacja na podstawie odcisków palców, geometrii dłoni (niekiedy także układu żył) czy też rozpoznawania siatkówki lub tęczówki oka. Istotnym aspektem w technikach biometrycznych jest test żywotności, który ma zabezpieczyć system przed oszukaniem go poprzez podstawienie zdjęcia, kopi silikonowej lub odciętej części organizmu. Do identyfikacji może służyć także geometria twarzy (niekiedy także ogólna geometria budowy ciała) - w zasadzie tylko te metody umożliwiają identyfikację bez współpracy rozpoznawanej osoby z zastosowaniem systemu CCTV.

Istotnym elementem systemów bezpieczeństwa są instalacje gaszenia - zarówno tradycyjne instalacje tryskaczowe, jak i przeznaczone głównie do pomiesczeń z sprzętem elektronicznym, rodzielni, itp systemy gaszenia gazowego. Wsród systemów tryskaczowych wyróżnia się instalacje suche (napełniane wodą dopiero po wykryciu pożaru, tylko takie mogą być stosowane w pomieszczeniach, w których występują ujemne temperatury) i instalacje mokre - cały czas nawodnione, a zaczynające działać dopiero po termicznym (lub znacznie rzadziej elektrycznym - z systemu detekcji pożaru) zniczeniu kapsułki blokującej wypływ wody przez dyszę. Typowe systemy wodne ze względu na duże ilości wypuszczanej wody (aczkolwiek scena znana z filmów że naskutek alarmu pożarowego wszystkie tryskacze w biurowcu urchamiają się jednoszczęnie jest fikcją - typowo działa tylko dysza która osiągneła temperaturę zadziałania) nie nadają się do stosowania w serwerowniach, rozdzielniach elektrycznych itp.

Wsórd systemów gazowych wyróżnia się instalacje oparte o gazy chemiczne i instalacje oparte o gazy obojętne. Zaletą gazów chemicznych typu Novec 1230[1346], FM-200[1347] (w stosunku co do gazów obojętnych - azotu, argonu, CO2 i ich miesznin) jest stosunkowo niewielka ilość butli, wymaganych przez system oraz nie wypieranie tlenu (brak problemów z oddychaniem, charakterystycznych głównie dla CO2), ich wadą jest duży koszt gazu oraz potencjalne ryzyko jego rozkłądu termicznego. Instalacje gaszenia gazowego wymagają odpowiedniej szczelności pomieszczenia i klap odciążających (umożliwienie wydstania nadmiaru powietrza i redukcji nadcisnienia powstałego na skutek uwolnienia gazu). Interesujące są także systemu utrzymujące stale obniżony poziom tlenu w pomieszczeniu (działają na zasadzie dopuszczania azotu) i uniemożliwiające w ten sposób spalanie płomieniowe

typowych substancji. Podobnie jak systemy gazowe wymagają one szczelności pomieszczenia (w odróżnieniu od nich cały czas, a nie tylko w trakcie akcji gaśniczej).

Jeżeli pomieszczenie które chemy zabezpieczać musi być intensywnie wentylowane (np. pomieszczenia silników spalinowych), a także w niektórych wypadkach dla serwerowni itp. pomieszczeń ciekawą alternatywą dla wyżej opisanych systemów są systemy mgły wodnej wytwarzanej w oparciu o wodę destylowaną - charaktryzyją się małą ilością wody potrzebną do gaszenia, a więc małymi stratami po pożarowymi (w porównaniu z tradycyjnymi tryskaczam p.poż.) i mogą być używane do zabezpieczania urządzeń elektrycznych.

Zachęcam także do zapoznania się z projektami centralki alarmowej, manipulatora do systemu alarmowego oraz stroboskopu LED mojego autorstwa. Zobacz też galerie zdjęć związanych z systemami alarmowymi[1348] oraz CCTV[1349].

Zobacz w Sieci: Systemy alarmowe i bezpieczeństwa: Elektronika - drzwi i okna[1350], Fire alarms[1351], Security cameras[1352], system aktywnego zapobiegania pożarom - OxyReduct®[1353], INERGEN® SYSTEM GAŚNICZY[1354], Stałe urządzenia gaśnicze gazowe cz.1[1355], Stałe urządzenia gaśnicze gazowe cz.2[1356], Stałe urządzenia gaśnicze gazowe cz.3[1357], Czy dwutlenek węgla jako środek gaśniczy powinien odejść do lamusa?[1358], Klimatyzacja w pomieszczeniach zabezpieczanych gazami gaśniczymi[1359].

Porady i wskazówki dotyczące projektowania i wykonawstwa

Szczególną uwagę należy zwrócić na etap projektowania i na nim upewnić się że realizowane są wszystkie przyjęte założenia, że projekty instalacji są wystarczająco szczegółowe i oparte na poprawnej inwentaryzacji podłączanych urządzeń. Dobrze przygotowany, rzetelny, szczegółowy i kompleksowy projekt ma kluczowe znaczenie dla realizacji obiektu i instalacji (zmiany i poprawki na jego etapie są dużo prostsze i tańsze niż na jakimkolwiek dalszym etapie realizacji inwestycji). W przypadku projektów instalacji elektrycznych należy upewnić się także że przeanalizowane są zwarciowości i selektywności i w oparciu o nie dokonany jest wybór aparatów i ich charakterystyk (analiza ta powinna zawierać stosowne obliczenia prądów zwarciowych, oraz wykresy charakterystyk czasowych). Należy też dopilnować aby projekt był na tyle elastyczny aby dopuszczał modyfikacje na etapie wykonania jak i eksploatacji obiektu / instalacji.

Bardzo istotne jest przygotowanie założeń stanowiących punkt wyjściowy do przygotowywania projektu. Zamieszczam przykładowe wytyczne dla projektu obiektu typu Data Center. Ponadto w przygotowywaniu oraz realizacji inwestycji budowlanych szczególną uwagę należy zwrócić na:

- szczegółowy projekt i specyfikacja wykonania poszczególnych prac (nie wolno zakładać że ekipa ma jakiekolwiek pojęcie o tym co ma zrobić - np. o kładzeniu izolacji przeciw-wodnej ani że kierować się będzie zdrowym rozsądkiem i logiką - np. wykona spadki na podłodze posiadającej kratki odpływowe oraz zrobi je we właściwym kierunku)
- szczegółowy plan kolejności robót uwzględniający kontrolę po każdym etapie i jego rygorystyczne przestrzeganie - wymagamy od wykonawcy sprawdzania poprawności wykonanych prac i dodatkowo sami sprawdzamy, jest to szczególnie ważne gdy wyniki danego etapu są zakrywane następnym (np. nie wolno pozwolić na układanie dalszych warstw podłogi zanim nie będzie wykonana całość izolacji przeciwwodnej w danym pomieszczeniu i nie zostanie ona sprawdzona, jak również nie wolno pozwolić na zasłonięcie tras wod.-kan. przed ich rzetelnym przetestowaniem)
- ciągły, rzetelny nadzór - do branży budowlanej pasują co najmniej 3 z 5 znanych zasad bezpieczeństwa komputerowego - Do not assume anything (Nie zakładaj niczego); Trust no-one, nothing (Nie ufaj nikomu ani niczemu); Paranoia is your friend (Paranoja jest Twoim przyjacielem)

Preferowaną konstrukcją budynku, wydaje się być wylewana na placu budowy monolityczna konstrukcja żelbetowa[1360] z słupami oraz miejscami ścianami nośnymi. Dobrze wykonana konstrukcja tego typu zapewnia solidność oraz duże możliwości aranżacji obiektu. Należy zadbać o stosowanie, wszędzie tam gdzie tylko jest to możliwe, materiałów niepalnych (zobacz też klasa reakcji na ogień[1361]). Jest to szczególnie istotne przy materiałach występujących w dużych ilościach, np. konstrukcyjnych, izolacyjnych, podłogowych, wykończeniowych, itp. (w szczególności należy unikać styropianu, itp na rzecz szkła piankowego[1362], wełny mineralnej[1363] lub szklanej[1364], preferować niepalne materiały wykończeniowe takie jak: szlifowany beton, cegła klinkierowa[1365], granit[1366] i inne skały magmowe[1367]). Ponadto należy zadbać o zapewnienie odpowiedniej wytrzymałości konstrukcji (także na obciążenia wyjątkowe[1368]) i separacji stref pożarowych (także przed przedostawaniem się dymu) poprzez zapewnienie odpowiedniej odporności

ogniowej[1369] konstrukcji, przegród budowlanych, przejść instalacyjnych (a najlepiej także redukowanie niepotrzebnych przejść pomiędzy strefami), itd. Nie można także zapominać o izolacji termicznej, akustycznej i przeciw-wodnej.

Obiekt powinien posiadać pełną, rzetelną dokumentację powykonawczą (szczególnie instalacji) umożliwiającą jej łatwą aktualizację przy wprowadzanych zamianach w trakcie użytkowania obiektu. Kable, rury itp. powinny być oznaczane w sposób trwały (odlepiające się po tygodniu naklejki odpadają) zgodny z tą dokumentacją - zarówno przy zakończeniu (gnieździe, podłączanym urządzeniu), jak i co pewien odcinek na swojej trasie (dopuszczalne jest zbiorowe oznaczanie wiązek kablowych).

Warto także na etapie projektu założyć stosowną redundancję oraz możliwości naprawy i modyfikacji poszczególnych systemów - przede wszystkim poprzez zapewnienie dostępu do nich (a gdy będzie on nie możliwy lub bardzo trudny poprzez położenie zapasowych przewodów). Dobrymi pomysłami wydaje się także:

- włożenie liniowych sond zalania w warstwę podłogi poniżej izolacji wodnej np. w warstwę płyt z wełny mineralnej
- umieszczenie liniowych sond zalania w przestrzeniach zakrytych narażonych na zalanie (np. ekrany w łazience, przestrzeń pod-wannowa, kanały tras wod-kan.)
- instalacje wod-kan. w ścianach powinny być zakrywane łatwo demontowanymi ekranami lub prowadzone natynkowo
- gdy nie można przeprowadzić tras wod.-kan. w kanale w ścianie to jeżeli jest to możliwe instalacje wod-kan. prowadzimy sufitem kondygnacji niższej (ew. maskując sufitem podwieszanym), a dopiero w ostateczności w podłodze
- stosowanie gwiazdy nie tylko w instalacjach elektrycznych i teletechnicznych, ale także na końcowych (prowadzonych poza pionami instalacyjnymi) fragmentach instalacji wodnej, kanalizacyjnej, itp. - rozwiązanie takie wydaje się szczególnie atrakcyjne gdy rury muszą być umieszczone np. w podłodze (i nie ma możliwości zapewnienia dostępu do nich) - pozwala na stosowanie na tym odcinku jednolitej elastycznej rury i uniknięcie jakichkolwiek łączeń w podłodze

Zasady projektowania systemów automatyki itp

- dzielimy system na moduły funkcjonalne i określamy interfejsy pomiędzy nimi - wybieramy standardowe, otwarte protokoły (i2c, rs485/modbus, ethernet/tcp-ip, …)
  - rekurencyjnie stosujemy tą zasadę do większych/bardziej złożonych bloków
  - protokoły komunikacyjne dobieramy z zapasem (przewidujemy pojawienie się kolejnych rejestrów, funkcji, … obciążenie sieci zakładamy na poziomie poniżej 30%)
- konsekwentnie trzymamy się określonych interfejsów, numerujemy wersje hardware i software, zachowujemy kompatybilność wsteczną (nowe urządzenie, nowa wersja muszą pracować w istniejącej sieci, nowa wersja musi w prosty sposób móc zastąpić poprzednią)
- sztywne trzymanie się standardowych interfejsów i protokołów jest szczególnie ważne w niskopoziomowych (trudno aktualizowalnych, debugowalnych, występujących w dużej liczbie egzemplarzy) urządzeniach
- dla transmisji danych stosujemy kable ekranowane, zostawiamy rezerwę pinów w modułach, nie kroimy modułów na miarę (lepiej mieć n+2 jednakowych modułów niż n każdy innego rodzaju)
- walczymy z problemem plątaniny kabli już na etapie założeń projektowych:
  - stosujemy modularność sterowania - np. 1 sterownik ethernet (lub conajmniej rs485) na niezbyt dużą grupę wejść (np. jeden panel przycisków) / wyjść (np. jeden rząd przekaźników)
  - nie żałujemy miejsca na PCB i funduszy na gniazdka przyłączeniowe (nie robić przylutowywanej na stałe wiązki kabli, pamiętać o masach i zasilaniach)
  - nie żałujemy miejsca na PCB na otwory montażowe (najlepiej na wszystkich modułach wykonywać je w identycznych miejscach - z myślą o przyszłej obudowie, a nie tam gdzie popadnie)
  - rozważyć stosowanie korytek grzebieniowych w szafach sterujących / sterownikach / etc

Przykładowe wytyczne dla projektu obiektu typu Data Center

Wymagania ogólne

1. Systemy zasilania i chłodu (w tym systemy ich automatyki, jeżeli występują na tym poziomie) nie mogą posiadać pojedynczych punktów awarii. Komplementarne (pod względem redundancji) podsystemy muszą mieć zapewnioną separację fizyczną (w tym pożarową).
2. Nośność stropów minimum 25kN/m2.
3. W pomieszczeniach technicznych i serwerowniach nie stosuje się sufitów podwieszanych.
4. Instalacja wody chłodniczej prowadzona w przestrzeni podpodłogowej. Przestrzeń ta posiada system detekcji wycieków oraz stosowne odpływy celem usunięcia ewentualnych wycieków czynnika.
5. Wysokość przestrzeni podpodłogowej zapewnia stosowną przestrzeń instalacyjną (w tym tą wymaganą dla komfortu prowadzenia prac serwisowych), nie mniej niż 1.2m.
6. Nośność podłogi minimum 25kN/m2, płyty metalowe lub gipsowe, ruszt skręcany z profili "C", zalecane jest stosowanie rusztu podłogi o zredukowanej liczbie podpór.
7. Opcjonalnie w przestrzeni podpodłogowej może się znaleźć sieć zasilająca (zwykła i gwarantowana) dla gniazdek ogólnego użytku montowanych w floorboxach oraz związana z nią sieć komputerowa. W takim wypadku floorboxy powinny być przyłączane wtyczkami (IEC 60309 16A + 2xRJ45) do gniazd zamontowanych na stałe do tras kablowych.
8. Przestrzeń główna minimum 2.5m od podłogi do najniższego elementu instalacji montowany do sufitu.
9. Całość instalacji zasilającej i IT do szaf serwerowych prowadzona górą (montowana do sufitu). Dla instalacji IT i niskoprądowych przewidziane 2 poziomy koryt siatkowych, jeden poziom koryt przewidziany dla kabli zasilających przyłączanych do szynoprzewodów. Przestrzeń na instalacje prowadzone górą minimum 1.2m.
10. Orientacyjny kształt i rozmieszczenie pomieszczeń, wraz z wymaganiami funkcjonalnymi i lokalizacją urządzeń infrastruktury zasilająco-chłodniczej przestawione jest w załączniku[1370].
11. W części birowo-socjalnej wydzielić pomieszczenia:
    - operatornia - minimum 3 stanowiska dla operatorów nadzorujących prace systemów skierowane w stronę ściany z monitorami głównymi, opcjonalnie z przeszkleniem (szyba akustyczna) na halę serwerów
    - warsztat machaniczny - frezarka numeryczna CRC, drukarka 3D, …
    - warsztat elektroniczny - stanowiska serwisu elektroniki (lutowanie, oscyloskop, zasilacz regulowany, …) - przykładowe[1371] rozwiązanie niewielkiego stanowiska serwisowego
    - węzeł socjalny i pomieszczenia wypoczynkowe
12. Zapewnić dogodną komunikację pomiędzy operatornią i warsztatami a serwerownią i pomieszczeniami technicznymi (w szczególności powinny znajdować się one w pobliżu)

System chłodniczy powinien być oparty na systemie wody lodowej, należy zapewnić w nim redundancje co najmniej n+1, wraz z redundancją rurociągów (2n) lub redundancję 2n / 2(n+1) wszystkich urządzeń (ułatwia to zapewnienie ich redundantnego zasilania). System chłodniczy powinien umożliwiać freecooling. Dopuszcza się możliwość realizacji redundancji poprzez stosowanie szaf klimatyzacyjnych z podwójnym obiegiem (podstawowy - woda lodowa, dodatkowy na potrzeby redundancji - freon). W tym wypadku należy zapewnić także redundancje w systemie wody lodowej na potrzeby urządzeń chłodzonych bezpośrednio cieczą.

Serwerownia oraz związane z nią pomieszczenia techniczne powinny być objęte systemami bezpieczeństwa takimi jak system kontroli dostępu, system sygnalizacji włamania i napadu, system telewizji dozorowej oraz system p.poż. Celem nadzorowania i rejestracji prac w serwerowni system monitoringu wizyjnego powinien nie posiadać martwych stref (monitorujemy wszystkie zimne i ciepłe korytarze, czyli przód i tył każdej z szaf rackowych) oraz powinien posiadać zduplikowaną rejestrację (lokalna i zdalna - poza serwerownią, z stałym nadzorem ich działania). Na system p.poż. powinien składać się system wykrywania pożaru wykorzystujący podsystem wczesnej, aktywnej detekcji dymu (zasysający), a także system aktywnego zapobiegania pożarom poprzez redukcję ilości tlenu w powietrzu i wspomagający go automatyczny system gaśniczy na gaz obojętny.

Konieczne jest także zapewnienie systemu nadzorującego warunki pracy (przede wszystkim temperaturę) takiego pomieszczenia, pracę systemu zasilającego i chłodniczego, a także sterującego pracą tych systemów (zwłaszcza systemu wytwarzania chłodu). System ten musi być zintegrowany z systemami informatycznymi (zastosowanie standardowych, otwartych protokołów komunikacji, wzajemna wymiana danych, itp.). System taki powinien także umożliwiać dokonywani wyłączenia części systemów w przypadku niemożliwości osiągnięcia poprawnych parametrów środowiskowych.

System zasilania

System zasilający planuje się wykonać w układzie redundancji N+2 (jedna jednostaka w gorącej, a jedna w zimnej rezerwie) w oparciu o zintegrowane z agregatami zestawy UPSów z masą wirującą (Diesel Rotary Uninterruptible Power Supply). Dzięki zastosowaniu odpowiedniego układu podłączeniowego (mieszanie stron poszczególnych sekcji serwerowni pomiędzy DRUPSami oraz opcjonalne zastosowanie systemu izolowano-równoległego) układ taki zapewnia brak pojedyńczego punktu awarii (jest odporny na awarię całej dowolnej sekcji zasilającej). Zapewnia to wymaganą na poziomie TIER IV redundancję systemu zasilania zrealizowaną na zaledwie N+1 jednostkach (zwykle w tym celu konieczne jest 2N jednostek). Kolejna jednostka (N + druga) zapewnia zachowanie pełnej redundancji w trakcie prowadzenia prac swerwisowych dowolnego elementu układu podstawowego (kiedy to zawsze wzrasta ryzyko wystąpienia dodatkowych problemów).

1. System zasilania gwarantowanego oparty jest o urządzenia typu DRUPS o mocy nie mniejszej niż 1670kVA.
2. Podstawowy system zrealizowany jest w redundancji N+1 z wykorzystaniem mieszania „stron" dystrybuowanych na serwerowni pomiędzy różnymi DRUPSami.
3. W systemie przewiduje się realizacje DRUPSa stanowiącego zimną rezerwę.
4. Główna dystrybucja zasilania z pojedynczego DRUPSa realizowana będzie w ramach modułu pomieszczeń technicznych na poziomie kondygnacji systemów energetycznych - np. w oparciu o szynoprzewód i montowane na nim skrzynki z zabezpieczeniami odpływowymi.
5. Dystrybucję zasilania na serwerowni planuje się z wykorzystaniem zestawów 4 szynoprzewodów o prądzie rzędu 630A, stanowiących 4 strony zasilania doprowadzane do urządzeń IT, umożliwia to bezproblemowe podłączanie typowych dwu zasilaczowych serwerów, jak równierz bardziej nietypowych urządzeń wymagających zasilania w trybie 3n, 3+1, itd.. Szynoprzewody prowadzone powinny być górą (pod sufitem) i prostopadle do kierunku głównej dystrybucji zasilania o której mowa powyżej
6. Układ zapewnia wymaganą standardem TIER IV separację fizyczną poszczególnych systemów zasilania poprzez umieszczenie urządzeń w osobnych separowanych pożarowo pomieszczeniach oraz stosowną separację tras szynoprzewodów służących do dystrybucji zasilania.
7. Wszystkie szynoprzewody oraz kable zasilające posiadają trwałe, czytelne, jednoznacznie identyfikujące przewód, miejsce do którego prowadzi, sposób zasilania (z którego DRUPSa, etc) oznaczenia na obu końcach i nie rzadziej niż co 5m na kablu/szynoprzewodzie.
8. Przewiduje się gwarantowane (bezprzerwowe) zasilanie całości systemu chłodu i klimatyzacji.

Okablowanie logiczne (niskoprądowe)

1. Okablowanie prowadzone wyłącznie w dostępnych przestrzeniach instalacyjnych w przygotowanych trasach kablowych.
2. Okablowanie prowadzone (głównie) w przestrzeni podsufitowej, w niewielkim stopniu dopuszcza się prowadzenie na trasach kablowych w przestrzeni podpodłogowej
3. Wydzielone trasy kablowe dla okablowania stałego (łączącego punkty dystrybucyjne oraz służacego systemom monitoringu i bezpieczeństwa) oraz okablowania dynamicznego (służacego połączeniom bezpośrednim w ramach komory serwerowej, podłączeniu sprzętów do najbliższych punktów dystrybucyjnych itd).
4. Przewiduje się podłączanie szaf rackowych dwoma redundantnymi linkami światłowodowymi z wykorzystaniem agregacji połączeń w ramach szafy (redundantny zestaw switch "top of the rack")

System monitorujący

Sieć monitoringu warunków środowiskowych i systemów bezpieczeństwa wykorzystuje okablowanie logiczne budynku oraz dedykowane switche w każdym z punktów dystrybucyjnych. Służy ona zapewnieniu komunikacji z monirorowanymi systemami (m.in. system p.poż, włamaniówka) i urządzeniami (m.in. DRUPSy, analizatory sieci elektrycznej, wytwornice wody lodowej) oraz kontrolerami/sterownikami sieci BMSowej (swobodnie programowalne sterowniki odczytujące stany wejść cyfrowych, analogowych itd). Sieć może wykożystywać dedykowane switche przemysłowe zbierające wszystkie połączenia z danego pomieszczenia.

Jako podstawę komunikacji między sterownikami przyjmujemy standardowe otware protokoły: BACnet/IP, Modbus TCP, SNMP. Monitorowane systemy muszą być w stanie przekazać przy pomocy sieci i któregoś z wymienionych protokołów pełną informację na temat swojego stanu (wraz z informacją na temat stanu swoich poszczególnych czujników).

1. System monitorujący korzysta z sieci opartej na ethernecie i TCP/IP. Jako podstawę komunikacji między sterownikami przyjmujemy standardowe otware protokoły: BACnet/IP, Modbus TCP, SNMP

2. Sieć monitrująca jest budowana w postaci redundantnej (dublowane połączenia sieciowe, ew. topologia pierścienia ethernetowego).
3. System alarmowy, kontroli dostępu i pożarowy wystawia pełną informację o swoim stanie (włącznie z tym które czujki PIR wykrywają ruch, która czujka pożarowa wzbudziła alarm itd) poprzez któryś z tych protokołów do sieci monitrującej. Wymagane jest rozwiązanie wspierające pełną funkcjonalność nadzorowanego systemu i dostarczone przez jego producenta lub przez niego certyfikowane. Rozwiązanie komunikacyjne nie może narzucać własnych (innych niż monitorowanego systemu) ograniczeń na ilość pozyskiwanej informacji z systemu (np. ilość odczytywanych zdalnie czujek p.poż).
4. System kontroli dostępu oparty jest na sterownikach drzwi bezpośrednio komunikujących się przez sieć monitorującą. Sterowniki posiadają autonomiczną pamięć dla danych dostępowych i logu zdarzeń. Sterowniki kontrolują stan otwarcia / zamknięcia drzwi, sygnalizują akustycznie niedomknięte drzwi, wystawiają ponadto tą informację do sieci monitorującej (ew. funkcjonalność tą może pełnić włamaniówka). System kontroli dostępu wykorzystuje m.in. dane biometryczne z czytników struktury żył w palcu dłoni lub w całej dłoni.
5. Monitorujemy parametry zasilania (analizatory z odczytem zdalnym modbus RTU lub inny otwarty protokół + zbramkowanie do TCP/IP), stan pracy DRUPSów, oraz stany wyłączników w rozdzielniach (każdy zespół DRUPS + rozdzielnie, ma własny sterownik zbierający danego z niego i wystawiający je do sieci monitrującej). Monitorowana jest wartość prądu na każdym z odpływów z szynoprzewodów (zarówno głównych, jak i dystrybucyjnych na serwerowni).
6. Urządzenia monitorujące podłączone do sieci odczytują za jej pośrednictwem parametry pracy innych urządzeń (detekcja zalania, systemy chłodzenia, wentykacji, ilość paliwa w zbiornikach, …).
7. System CCTV oparty jest na kamerach IP z protokołem RTSP, sprzętową detekcją ruchu, obsługą i konfiguracją poprzez wywołania CGI lub interfejs CLI.
8. Awaria systemu lub sieci monitorującej nie może wpływać na pracę nadzorowanych przez nią systemów:
   - bezpieczeństwa - włamaniówka, pożarówka, kontrola dostępu
   - zasilania
   - chłodzenia (w przypadku braku sterowania centralnego układ powinien móc funkcjonować w oparciu o wewnętrzną automatykę poszczególnych urządzeń)
9. Sieć monitorująca ma postać modularną i rozproszoną. Połączenia pomiędzy modułami realizowane są poprzez łącza optyczne (światłowody), co zapewnia gwarancję separacji galwanicznej poszczególnych fragmentów sieci monitorującej. Na przykład: całość systemu monitorującego pomieszczenie DRUPS (w tym systemy czytające stany położeń wyłączników w rozdzielniach itd.) jest w tym pomieszczeniu i jest wpięta do switcha ethernetowego znajdującego się w tym pomieszczeniu, który posiada redundantne światłowodowe połączenia zresztą sieci.
10. W każdym module sieci należy przewidzieć 20%, lecz nie mniejszą niż 2 ilość wolnych wejść/wyjść (np. portów w switchach ethernetowych, wejść w modułach DI, …).
11. System zarządzający zapewnia zdalny dostęp do wszystkich odczytywanych parametrów. Dostęp realizowany poprzez protokół HTTP, z nielimitowaną ilością równoczesnych użytkowników. Dostęp musi funkcjonować poprawnie, niezależnie od używanej przeglądarki, systemu operacyjnego, itd. pod warunkiem iż przeglądarka jest zgodna z standardem HTML 4.0 i protokołem HTTP 1.1.
12. Wszystkie kable w systemach automatyki i monitoringu posiadają trwałe, czytelne, jednoznacznie identyfikujące kabel, jego funkcję i miejsce do którego prowadzi oznaczenia na obu końcach i nie rzadziej niż co 5m na kablu.
13. System zapenia logowanie (archiwizację) monitorowanych parametrów np. w systemie Zabbix.

Powyższe wytyczne opierają się na założeniach że:

1. w obiekcie są autonomiczne systemy bezpieczeństwa (p.poż, włamaniówka), które są tylko wpięte (pod względem monitoringu) do sieci monitorującej
2. w sieci monitorującej (abstrachując od tego czy będzie to jedna sieć czy coś zostanie wydzielone VLANami czy nawet sprzętowo) działają też systemy kontroli dostępu i monitoringu wideo

W związku z tym nie ma silnego rozdzielenia systemów bezpieczeństwa od monitoringu środowiskowego. Można tak jednakzrobić (co ma zalety i wady). Natomiast wydzielony (jako autonomiczna jednostka) z systemu monitoringu został system sterowania układem chłodu (taka separacja umożliwia jego niezależne stworzenie specjalistom od automatyki chłodnictwa).

Systemy przeciw-pożarowe

1. wszystkie pomieszczenia techniczne (być może za wyjątkiem trafo i rozdz. SN gdyby miały mieć wentylacje powietrzem zewnętrznym) objęte są systemami:
   - aktywnej detekcji dymu (typu titanus, vesda)
   - systemem gaszenia gazowego (gaz obojętny lub Novec 1230)

    - systemem konwencjonalnych czujek dymu (system musi być wstanie wskazać która czujka wywołał alarm)
2. system aktywnej detekcji dymu używany jest do:
    - informowania obsługi obiektu o prawdopodobnym zagrożeniu (wydzielona sygnalizacja alarmu technicznego)
    - sterowania wentylacją (np. wyłączenie freecoolingu powietrznego gdy w zasysanym powietrzu wykryty dym)
3. do sterowania systemem gaszenia używana jest koincydencja dwóch czujek konwencjonalnych lub czujki konwencjonalnej i aktywnej detekcji dymu
4. centrala p.poż musi umożliwiać proste sterowanie z panelu operatorskiego:
    - zamknięciem wybranych klap dymnych, wyłączeniem funkcji wentylacyjnych
    - blokowaniem czujek i stref
    - blokowaniem sygnalizacji optyczno-akustycznej
    - blokowaniem urządzeń transmisji alarmu do PSP

    żadan z powyżej opisanych blokad nie może skutkować blokowaniem działania klap pożarowych, komunikacji z siecią monitorującą, itd
5. pomieszczenia w których trudne (np. pomieszczenia silnie wentylowane) / niepotrzebne (np. drogi komunikacyjne) byłoby zastosowanie SUG gazowego objęte systemem wysokociśnieniowej mgły wodnej
6. wybrane pomieszczenia techniczne zabezpieczane są dodatkowo atmosferą ochronną - mają zredukowany poziom tlenmu poniżej granicy palności (rozwiązania typu OxyReduct) - należy przewidzieć miejsce na montaż takich urządzeń oraz odpowiednią szczelność pomieszczeń

Centrala SSP powinna zapewniać co najmniej (szczegółowe warianty alarmowania, reagowania itd powinny być dostosowane do potrzeb danego obiektu):

- pełna komunikacja/monitoring z wykorzystaniem MODBUS lub BacNet, umożliwiający wizualizację stanu poszczególnych czujek oraz stanu stref
- przypisanie urządzeniom wyjściowym (sterowanym z centrali) stref i typów, wyróżniając następujące grupy urządzeń:
    1. urządzenia "zabezpieczenia wentylacyjnego" (klapy pożarowe, przekaźniki central wentylacyjnych, itp mające na celu uniemożliwienie rozprzestrzeniania się dymu i ognia pomiędzy strefami); brak przypisania do stref
    2. sygnalizatory optyczno-akustyczne (bez względu na sposób podłączenia sygnalizatora - stało prądowy, adresowalny, ...); przypisane do stref, mogą mieć flagę ogólne
    3. urządzenia wytwarzające atmosferę ochronną (redukcja tlenu w powietrzu); przypisane do stref, mogą być współdzielone
    4. urządzenia transmisji alarmu do straży pożarnej/centrali monitoringu; brak przypisania do stref
    5. urządzenia gaśnicze (gaszenie gazowe, ...); przypisane do stref, mogą być współdzielone
    6. urządzenia oddymiające; przypisane do stref, mogą być współdzielone
- wyróżniać następujące stany pracy wraz z możliwością przypisania do nich różnych akcji (przykłady podane):
    1. DOZOROWANIE
    2. USTERKA
    3. ALARM WSTĘPNY
        - wysterowanie urządzeń "zabezpieczenia wentylacyjnego"
        - jeżeli strefa posiada urządzenia grupy wytwarzające atmosferę ochronną to zwiększenie ich wydajności (wejście w tryb pożarowy) oraz uruchomienie sygnalizacji optyczno-akustycznej w strefie
    4. ALARM 1 STOPNIA
        - działania trybu "ALARM WSTĘPNY"
        - uruchomienie sygnalizacji optyczno-akustycznej w całym obiekcie
        - wysterowanie urządzeń oddymiających w tej strefie
    5. ALARM 2 STOPNIA
        - działania trybu "ALARM 1 STOPNIA"
        - wysterowanie urządzeń grupy transmisji alarmu (powiadomienie straży)
    6. GASZENIE
        - działania trybu "ALARM 2 STOPNIA"
        - jeżeli strefa posiada urządzenia gaśnicze to ich wysterowanie (rozpoczęcie gaszenia)
- umożliwiać realizowanie rozbudowanych i swobodnie programowanych schematów alarmowania - np:
    1. zadziałanie pojedynczego/pierwszego detektora w strefie:
        - ALARM WSTĘPNY w strefie
        - reset detektora po 5 minutach
            - gdy brak ponownego alarmu przez 60s -> przejście w tryb DOZOROWANIE
            - gdy ponowny alarm w przeciągu 60s -> przejście w tryb ALARM 1 STOPNIA
    2. zadziałanie drugiego detektora w strefie (jeżeli jest to kolejny detektor aktywny lub drugi poziom detektora aktywnego):

    - podniesienie trybu działania do "ALARM 1 STOPNIA" dla danej strefy
  3. zadziałanie drugiego detektora w strefie (jeżeli co najmniej jeden z detektorów to detektor konwencjonalny):
     - podniesienie trybu działania do "GASZENIE" dla danej strefy
  4. brak reakcji ochrony na tryb ALARM 1 STOPNIA w przeciągu 5 min:
     - przejście w tryb ALARM 2 STOPNIA
  5. brak reakcji ochrony na tryb ALARM 2 STOPNIA w przeciągu 10 min:
     - przejście w tryb GASZENIE
- jedno-przyciskowe funkcje
  - poziom dostępu 0:
    - WYSTEROWANIE urządzeń grupy A - uruchomienia trybu pożarowego
    - WYCISZENIE sygnalizacji optyczno-akustycznej (urządzenia grupy B), w przypadku otrzymania kolejnej informacji z czujki o pożarze sygnalizatory zostają ponownie wysterowane
  - poziom dostępu 1:
    - BLOKADA urządzeń grupy B
  - poziom dostępu 2:
    - BLOKADA urządzeń grupy D
    - BLOKADA urządzeń grupy E
- możliwość blokady poszczególnych detektorów oraz całych stref
- funkcje start/stop gaszenia realizowane z przycisków w strefie objętej gaszeniem której dotyczą

Systemy bepieczeństwa

Centrala SSWiN powinna zapewniać co najmniej:

- pełna komunikacja/monitoring z wykorzystaniem MODBUS lub BacNet, umożliwiający wizualizację stanu poszczególnych czujek oraz stanu uzbrojenia, alarmowania stref
- możliwość załączenia systemu z aktywną czujka mającą opóźnienie (cas na wyjście) - jeżeli po upływie czasu na wyjście czujka nadal aktywna to alarm
- współpraca z systemem kontroli dostępu (pin rozbrajający alarm musi się zgadzać z użytkownikiem karty użytej do otwarcia drzwi)

System sterujący pracą układu chłodu

1. wystawia pełne danych o parametrach systemu chłodu (temperatury, ciśnienia, przepływy, załączone podzespoły, wilogotność powietrza ...) do sieci monitorującej
2. umożliwia określanie z sieci monitorującej preferowanych wartości lub widełek wydajności sterowanych urządzeń. Urządzenie wykonawcze utrzymuje swoją wydajność w zadanym zakresie, bez względu na to iż jego automatyka sugerowałaby wyjście poza ten zakres, dopóki nie zostaną przekroczone poziomy krytyczne parametrów pracy.
3. w przypadku braku działania systemu monitorującego lub po przekroczeniu zadanego progu alarmowego system musi mieć możliwość zignorowania ustawionej preferencji i pracy autonomicznej (związane jest to z wymogami dotyczącymi zachowania pracy układu chłodu w przypadku awarii sieci monitorującej)
4. system musi zapewnić odpowiednią redundancje
5. w systemie preferowane są otwarte protokoły komunikacje (takie o których mowa w opisie sieci monitorującej) pomiędzy jego sterownikami a urządzeniami, itd.
6. System automatyki chłodu jest jednym z elementów nadzorowanych przez system monitoringu.
7. System automatyki chłodu może wykorzystywać do komunikacji z urządzeniami wykonawczymi i czujnikami sieć systemu monitoringu.

Baza przydatnych programów

W artykule tym postaram się omówić najważniejsze grupy programów komputerowych, będę również odwoływał się do konkretnych ich przedstawicieli. Przy wyborze przedstawicieli podstawowymi kryteriami były: wolność danego programu (musi być Free Software), popularność (musiałem mieć z nim doczynienia) oraz to czy mi przypadł do gustu (nie mogę go nie lubić; programy które szczególnie przypadły mi do gustu wyróżnione są **pogrubieniem**).

Zachęcam też do odwiedzenia Katalogu Wolnego Oprogramowania[1372] na stronach FSF, Kategoria:Wolne i Otwarte Oprogramowanie[1373] na stronach Wikipedii oraz Linux Software[1374], a także zbioru programów i skryptów dla KDE: kde-apps.org[1375].

W poniższym spisie nie są ujęte informacje o:

- standardowych (i trochę mniej standardowych) narzędziach shellowych
- programach do komunikacji z wykorzystaniem sieci komputerowych (zarówno po stronie serwera jak i klienta)
- konfiguracji sieci
- oprogramowaniu inżynieryjnym

System operacyjny

Najważniejszym elementem oprogramowania każdego komputera jest system operacyjny. System operacyjny jest tym elementem który pośredniczy między sprzętem a oprogramowaniem i często realizuje wiele funkcji w imieniu tego drugiego. W jego skład (prócz jądra i wchodzą też podstawowe programy użytkowe, o których jednak w następnych punktach.

Najprostszym stosowanym do dzisiaj systemem operacyjnym jest Disk Operation System, czyli znany DOS, odradzający się ostatnio pod postacią FreeDOS'a. Jest to jednoużytkownikowy, jednozadaniowy (niekiedy może być to ogromną zaletą) mały system operacyjny o bardzo niskich wymaganiach sprzętowych. W związku ze swoją specyfiką system ten nie jest obecnie popularny, jednak jest bardzo użyteczny w specyficznych zastosowaniach.

Trochę po przeciwnej stronie plasują się Unix'owate systemy operacyjne takie jak GNU/Linux czy też *BSD, są to wieloużytkownikowe, wielozadaniowe systemy operacyjne, posiadające większe (ale nadal nie duże) wymagania systemowe. Są to popularne systemy ogólnego przeznaczenia - stosowane zarówno jako serwery sieciowe, komputery biurowe, domowe i profesjonalne maszyny multimedialne.

- Linux[1376] - znany system operacyjny … a tak naprawdę tylko jądro …
- FreeBSD[1377] - system z rodziny BSD
- NetBSD[1378] - kolejny system z rodziny BSD
- OpenBSD[1379] - i jeszcze jeden system z rodziny BSD
- FreeDOS[1380] - wolna wersja DOS'a, binarnie zgodna z tymi które co poniektórzy jeszcze pamiętają … interesująca jest też jego dystrybucja typu Live - Balder[1381]
- HaikuOS[1382] - znany również jako OpenBeOS wolny odpowiednik BeOS'a, niestety ciągle nie gotowy …
- Syllable[1383] -
- AtheOS[1384] -
- IdyllaOS[1385] -
- wine[1386] - implementacja API Windows dla środowiska X11
- dosbox[1387] - emulator DOSu
- bochs[1388] - emulator komputera PC
- qemu[1389] - emulator komputera PC

Środowisko graficzne

Ważnym elementem każdego popularnego (w sensie grupy docelowej, a nie rzeczywistej popularności) systemu operacyjnego jest środowisko graficzne. Prawie zawsze następuje rozdzielenie samego silnika graficznego (obsługującego wyświetlanie tak jak xFree, czy X.org) od graficznego odpowiednika opisywanego powyżej shell'a (tworzącego przyjazne użytkownikowi środowisko do uruchamiania programów i zarządzania plikami ale również odpowiedzialnego za zarządzanie oknami), takiego jak Gnome, KDE (obydwa zawierają bogactwo dodatkowego oprogramowania - managery plików, przeglądarki WWW, edytory, klienci poczty, …), bardzo konfigurowalnego Enlightenment oraz mających zdecydowanie mniejsze wymagania niż dwa pierwsze Window Maker, czy XFce.

W tym miejscu trzeba też wspomnieć o graficznych interfejsach i odpowiednikach podstawowych narzędzi tekstowych (takich jak archiwizery).

- X.org[1390] - aktualnie chyba najpopularniejszy X-serwer
  Debian: xorg[1391] xserver-xorg[1392]
- xfree86[1393] - starsza gałąź X-serwera
- GNOME[1394] - popularne śrdowowisko graficzne
- KDE[1395] - rozbudowane i bardzo popularne śrdowowisko graficzne, zawierające wiele przydatnego oprogramowania

- kdm[1396] - menager logowania KDE
- gdm[1397] - menager logowania GNOME
- klipper[1398] - narzędzie schowka dla KDE
- clipit[1399] - narzędzie schowka
- lxpanel[1400] - panel (menu, pasek zadań, zegar, ...)
- fbpanel[1401] - panel (menu, pasek zadań, zegar, ...)
- tint2[1402] - panel (menu, pasek zadań, zegar, ...)
- knotes[1403] - żółte karteczki dla KDE
- openbox[1404] - menager okien
- fluxbox[1405] - menager okien
- ratpoison[1406] - menager okien obsługiwany wyłącznie przy pomocy klawiatury
- Enlightenment[1407] - menager okien / lekkie śrdowowisko graficzne
- Window Maker[1408] - menager okien / lekkie śrdowowisko graficzne
- XFce[1409] - lżejsze popularne śrdowowisko graficzne
- wmctrl[1410] - narzedzie do sterowania menagerem okien
- xosd[1411] - narzędzie do wyświetlania informacji poprzez X serwer w trybie OSD
  Homepage: `http://ignavus.net/software.html`
  Debian: xosd-bin[1412]
- xscreensaver[1413] - zbiór wygaszaczy ekranu wraz z narzędziami do ich konfigurowania i uruchamiania

Menażery plików, emulatory terminali, aplety menu

Swego rodzaju odpowiednikami shelli dla środowiska graficznego są menażery plików, emulatory konsol oraz różnego rodzaju panele, aplety, systemy menu działające na pulpicie

- konqueror[1414] - standardowy menażer plików w KDE, obsługuje wiele protokołów dostępu do plików (np. sftp)
- tuxcmd[1415] - rozbudowany manager plików
- Krusader[1416] - kolejny manager plików
- KRename[1417] - narzędzie do zmieniania nazw plikół
- ark[1418] - graficzny zarządca archiwów z KDE
- kfilereplace[1419] - narzędzie do wyszukiwania i zamiany dla KDE
- kfind[1420] - wyszukiwacz plików dla kde
- p7zip[1421] - rozbudowany archiwizer/kompresor plikó
  Debian: p7zip-full[1422]
- konsole[1423] - standardowy emulator terminala w KDE
- aterm[1424] - inny popularny emulator terminala
- tilda[1425] - przezroczysty emulator terminala bez widocznych elementów okna

Edytory tekstu

Kolejną bardzo ważną grupą programów są wszelkiego rodzaju edytory tekstu, zarówno te działające w trybie tekstowym jak Vim i jego klony, Emacs, Pico, jak i te proste notatniki trybu graficznego (Kedit, ...) aż do rozbudowanych procesorów tekstu wchodzących w skład pakietów biurowych (i omawianych w tamtej części). Oprócz wprowadzania i poprawiania tekstu posiadają one wiele ułatwiających życie funkcji - począwszy od zawijania linijek, poprzez kolorowanie składnia, automatyzację wcięć w kodzie, ...

- `vim` - legendarny edytor tekstowe

  podstawowe komendy VIMa: `i` - tryb wstawiania, `Insert` - zmiana trybu wstawiania i zastępowania, `Esc` - wyjęscie z trybu wstawiania/zastępowania, `y` - kopiuje, `d` - kasuje (po `y`, `d` mozna podac np `20l` lub `20{strzalka w prawo}` co oznacza 20 kolejnych znakow, `w` oznacza slowo, podwojenie `y` lub `d` oznacza kopiowanie/kasowanie całej linii), `x` - kasuje tyle znakow jaka liczba poprzedzony, `p` - wkleja po, `P` - wkleja przed, `/` - szukanie, `N` - poprzednie znaleznien, `n` - nastepne, `1,$s#szukany#zamiennik#g` - zastępowanie w całym pliku (bez adresu `1,$` - w bierzącej linii), `:w` - zapis `:q` - wyjście `:q!` - wujście bez zapisywania `:wq` - zapis i wyjście `:next` - następny plik `:previous` - poprzedni plik `:split` - podział okna `Ctrl+W` - przełączanie między oknami

Ściąga do vi (pl)[1426], Ściąga do vi (en)[1427] - kilkustronicowe broszurki z najważniejszymi poleceniami, graficzny tutorial do vim'a[1428] (i alternatywna wersja[1429]), Praktyczny Vim[1430]
Warto także zainteresować się sprawdzaniem pisowni w vim: instalujemy paczkę vim-spellfiles-pl (jest w experimental, instaluje się w `/usr/share/vim/vim70/spell/`), w `~/.vimrc` wpisuje: `set spelllang=pl,en`, a następnie sprawdzamy przy pomocy `:set spell`.
Homepage: http://www.vim.org/

— `emacs` - inny kultowy edytor tekstowy
Ściąga do Emacsa[1431] - kilkustronicowa broszurka z najważniejszymi poleceniami
Homepage: http://www.gnu.org/software/emacs/emacs.html
— XEmacs[1432] -
— vile[1433] - czyli próba pogodzenia vi i emacs'a
— kwrite[1434] (kate) - edytor z kolorowaniem składni i innymi bajerami dla KDE
Homepage: http://kate.kde.org/

Systemy DTP

Są to systemy umożliwiające przygotowanie publikacji zawierającej sformatowany tekst (w tym często wyrażenia matematyczne, ...), połączony z elementami graficznymi, tabelami, ... czyli tego wszystkiego co widzimy w książkach, czasopismach i gazetach. Tu również istnieje szeroka pula programów począwszy od systemów wyłącznie przetwarzających podane mu zwykłe pliki tekstowe (z odpowiednimi komendami sterującymi) oraz pliki graficzne w gotową publikację (słynny system TeX i jego rozwinięcie LaTeX) aż do systemów stanowiących zintegrowane środowisko do przygotowania publikacji (Scribus). Co ważne oba te rozwiązania mają istotne zalety - wygoda czystych plików tekstowych oraz wygoda zintegrowanego graficznego środowiska programu.

— texlive[1435] (LaTeX) - dystrybucja systemu składu tekstu TeX / LaTeX
Homepage: http://www.latex-project.org/
— Scribus[1436] - graficzne środowisko DTP
— pdfedit[1437] - edytor plików PDF
— flpsed[1438] - edytor umożliwiający dodanie dodatkowego tekstu do plików ps
— Xournal[1439] - program umożliwiający dodawanie notatek na plikach pdf
— pdftk[1440] - zestaw narzędzi do manipulowania plikami pdf (np. dzielenia, łączenia itd)
— ghostscript[1441] - system obsługi PostScriptu i zastaw narzędzi do plików postscriptowych (m.in. ps2pdf, ps2ps, gs, ...)
— psutils[1442] - zastaw narzędzi do plików postscriptowych (m.in. pstops, umożliwiający wraz narzędziami z innych pakietów np przeskalowanie strony A4 z pliku pdf do formatu A3 - `pdftops $f.pdf - | pstops -w 420mm -h 297mm -p A3 0@1.4142 | ps2pdf -sPAPERSIZE=a3 - $f-A3.pdf`, umieszczenie 4 stron A4 na jednej kartce - `pstops "4:0@.5(0,14.85cm)+1@.5(0,0)+2@.5(10.5cm,14.85cm)+3@.5(10.5cm,0)" infile.psoutfile.ps`)
— poster[1443] - przetwarzanie plików PS celem drukowania na wielu stronach
— fontforge[1444] - edytor fontów PostScript'owych
— fontmatrix[1445] - menager czcionek
— (gwaterfall) - przeglądarka czcionek
— Xpdf[1446] - przeglądarka pdf'ów oraz zestaw narzędzi do ich obsługi (np. `pdftotext -enc UTF-8 -nopgbrk input.pdf` przekonwertuje pdf na text czy też pdftops konwertujący PDF do PostScriptu)
— kpdf[1447] - przeglądarka pdf'ów dla KDE
— evince[1448] - przeglądarka PS/PDF
— GV[1449] - przeglądarka plików ps i pdf
— dvipng[1450] - program do konwersji plików DVI na PNG
— highlight[1451] - nardzędzie do generowania plików latex, html, ... z kodem programu z pokolorowaną składnią obsługujace wiele języków programowania
— html2ps[1452] - narzędzie do konwersji HTML na PS
— fbgs[1453] - konsolowa oparta na framebuferze przeglądarka PS/PDF
— SVP[1454] - konsolowa (nie wymagająca X'ów) przeglądarka PDF

Pakiety biurowe i ich składniki

Kolejną grupą programów operujących głównie na tekście są tak zwane pakiety biurowe, w ich skład wchodzi przedewszystkich procesor tekstu (edytor o rozbudowanych opcjach formatowania oraz operujący głównie na własnym formacie pliku, obsługuje eksport i import z formatów obcych - AbiWord) oraz arkusz kalkulacyjny (program który wykonuje operacje matematyczne, statystyczne, na komórkach tabeli, która jest podstawą jego działania), programy te niemal zawsze funkcjonują w trybie graficznym (ale są wyjątki). Oprócz tych programów często znajdziemy edytory grafiki i programy do tworzenia prezentacji.

- OpenOffice[1455] -
- LibreOffice[1456] - fork OpenOffice'a
- Calligra[1457] - dawniej KOffice
- AbiWord[1458] - (w zasadzie część GNOME)
- Gnumeric[1459] - (w zasadzie część GNOME)

Edytory graficzne

Programy do edycji graficzne dzielimy zasadniczo na trzy typy - programy do tworzenia grafiki rastrowej, grafiki wektorowej i animacji. Grafikę rastrową używa się do obrazków o nieregularnych kształtach, zdjęć, ..., umożliwia ona stosowanie licznych filtrów i efektów, w śród programów do jej edycji trzeba wymienić słynnego Gimpa. Z kolei podstawową zaletą grafiki wektorowej jest niezmienność jakości przy powiększaniu oraz zdolność do opisywania struktury (linia jest obiektem, a nie tylko zbiorem punktów), dlatego grafika ta jest używana głownie do tworzenia symboli i log oraz schematów. Z kolei animacje tworzone są jako sekwencyjnie wyświetlane pliki graficzne (z ewentualnymi efektami przejścia).

- **Grafika rastrowa**
  - GIMP[1460] - nie trzeba przedstawiać ;-)
  - Kolour Paint[1461] - prosty ale bardzo użyteczny program graficzny
  - krita[1462] - rozbudowany edytor grafiki rastrowej dla KDE
  - cinepaint[1463] - edytor grafiki rastrowej
  - - edytor grafiki rastrowej
  - tgif[1464] - edytor grafiki rastrowej
  - tkpaint[1465] - edytor grafiki rastrowej
- **Grafika 3D**
  - blender[1466] - program do grafiki 3D
  - YafRay[1467] - program do grafiki 3D
  - k3d[1468] - program do grafiki 3D
  - kpovmodeler[1469] - program do modelowania 3D
  - sced[1470] - program do grafiki 3D
  - vertex[1471] - program do grafiki 3D
  - Moonlight3D[1472] - program do grafiki 3D
  - ayam[1473] - program do grafiki 3D
- **Grafika wektorowa**
  - inkscape[1474] - odpowiednik GIMPa dla grafiki wektorowej ;-)
  - karbon[1475] - rozbudowany edytor grafiki wektorowej dla KDE
  - xfig[1476] - edytor grafiki wektorowej
  - graphviz[1477] - system wizualizacji grafów
  - pythoncad[1478] - CAD
  - qcad[1479] - CAD, @ Wikipedia[1480]
  - dia[1481] - edytor diagramów
  - ipe[1482] - edytor grafiki PDFu)
- **Narzędzia graficzne**
  - imagemagick[1483] - pakiet do obróbki grafiki
  - netpbm[1484] - zestaw narzędzi do konwersji i obróbki plików graficznych, bazujących na wygodnym raw'owym formacie grafiki (niejako analogicznym do cdaudio) - Portable anymap[1485]
  - przykłady[1486] -)
  - kover[1487] - edytor okładek CD
  - gtkmorph[1488] - morphing
  - autotrace[1489] - bitmap to vector

— potrace[1490] - bitmap to vector
— pstoedit[1491] - PS, PDF to vector
— gozer[1492] - text renderer
— pngcrush[1493] - optymalizacja PNG),
— digikam[1494] - manager zdjęć
— galrey[1495] - tworzenie galerii zdjęć w HTMLu
— gocr[1496] - ORC
— gtkam[1497] - obsługa aparatów cyfrowych
— metapixel[1498] - generator mozaiek fotograficznych
— panorama[1499] - zdjęcia panoramiczne
— xaos[1500] - generator fraktali
— xsane[1501] - obsługa skanera
— kiconedit[1502] - edytor ikon z KDE
— ksnapshot[1503] - program do tworzenia zrzutów ekranu dla KDE
— gphoto2[1504] - narzędzie do obsługi aparatów fotograficznych niezgodnych z USB mass storage device class[1505]
— kcoloredit[1506] - edytor kolorów kde, pozwala m.in. na złaopanie koloru z dowolnego elementu na ekranie)

Edytory wideo

Programy te umożliwiają obróbkę i montaż cyfrowego wideo wraz z jego ścieżką audio, dodawanie efektów przejść, elementów animacji, a także kompresję i konwersję między formatami. posiadają również moduł umożliwiający przechwytywanie wideo (jego ucyfrawianie).

— cinelerra[1507] - rozbudowany edytor wideo (program do montarzu AV),
— kino[1508] - rozbudowany edytor wideo (program do montarzu AV)
— kdenlive[1509] - rozbudowany edytor wideo (program do montarzu AV)
— openmovieeditor[1510] - rozbudowany edytor wideo (program do montarzu AV)
— openshot[1511] - edytor wideo
— avidemux[1512] - edytor wideo,
— PITIVI[1513] - edytor wideo
— LiVES[1514] - edytor wideo,
— cinefx[1515] - odtwarzacz, edytor efektów wideo, ...,
— zoneminder[1516] - rozbudowany system nadzorczy dla CCTV
— streamer[1517] - rejestrator video
— dvgrab[1518] - rejestrator video
— motion[1519] - rejestrator video z detekcją ruchu
— mvc[1520] - rejestrator video z detekcją ruchu
— transcode[1521] - program do konwersji, rejestracji i przetwarzania AV
— MEncoder[1522] - program do konwersji, rejestracji i przetwarzania wideo
— Gv4l[1523] - graficzny interfejs do transcode (rejestracja wideo z podglądem ...
— dvbcut[1524] - graficzne narzędzie do przycinania strumieni DVB / MPEG
— theorur[1525] - streamer video,
— OGMtools[1526] - zestaw narzędzi do obsługi formatu ogg,
— MKVToolnix[1527] - zestaw narzędzi do obsługi formatu Matroska,
— Konverter[1528] - graficzna nakładka na MEncoder dla KDE,
— mjpegtools[1529] - narzędzia do obsługi formatu M-JPEG,
— vcdimager[1530] - narzedzie do tworzenia i rippowania VideoCD,
— vorbis-tools[1531] - narzędzia do obsługi Ogg Vorbis,
— - narzędzia do obsługi Ogg Vorbis,

Edytory audio

Oprogramowanie to umożliwia montaż, doadawanie efektów, konwersję i kompresję oraz przechwytywanie audio.

— audacity[1532] - edytor audio
— ecawave[1533] - edytor audio
— ardour[1534] - edytor audio

— gnusound[1535] - edytor audio
— quelcom[1536] - edytor audio
— rezound[1537] - edytor audio
— snd[1538] - edytor audio
— sweep[1539] - edytor audio
— wavesurfer[1540] - edytor audio
— - edytor audio
— glame[1541] - edytor audio
— mhWaveEdit[1542] - edytor audio
— SoniK[1543] - edytor audio
— Kwave[1544] - edytor audio
— Rosegarden[1545] - sequencer MIDI
— Wavemixer[1546] - edytor audio
— arecord[1547] - np. `arecord -fcd `date +"%F_%T.wav"`` zapisze wybrane w mikserze źródło przechwytywania do pliku w jakości CD
— krecord[1548] - rejestrator audio
— sound-recorder[1549] - rejestrator audio
— kstreamripper[1550] - rejestrator audio
— dtmfdial[1551] - generator sygnałów tonowych
— synaesthesia[1552] - wizualizacja audio
— mpgtx[1553] - pliki MPEG
— easytag[1554] - edytor tagów ID3
— prokyon3[1555] - manager mp3
— normalize[1556] - standaryzuje głośność pliku wav
— TiMidity[1557] - MIDI to wav
— sox[1558] - audio convert
— LAME[1559] - mp3 encoder
— darkice[1560] - audio streamer
— kaudiocreator[1561] - frontend do rippowania CD dla KDE
— kmid[1562] - odtwarzacz MIDI dla KDE
— cdparanoia[1563] - czytnik audioCD

Przeglądarki multimedialne i odtwarzacze

Przeglądarki multimedialne i odtwarzacze audio/video są programami służącymi do wyświetlania / odtwarzania plików w wielu formatach, odtwarzacze potrafią zapisywać tak zwane "playlisty" czyli listy plików które mają odtwarzać, przeglądarki oferują tworzenie podglądów katalogów, przeglądać w trybie pokazu slajdów oraz dokonywać prostych modyfikacji plików graficznych i zmian formatu zapisu.

— VLC media player[1564] - uniwersalny odtwarzacz z projektu VideoLAN
— MPlayer[1565] - uniwersalny odtwarzacz multimedialny
— mpv[1566] - fork MPlayer'a
— MPlayer2[1567] - fork MPlayer'a
— kmplayer[1568] - odtwarzacz multimedialny dla KDE potrafiący korzystać z silników GStreamer, MPlayer, Xine i Ice Ape
— omxplayer[1569] - odtwarzacz multimedialny potrafiący wykorzystać wsparcie sprzętowe dekodowania wideo na Raspberry Pi
— xine[1570] - odtwarzacz multimediów
Debian: xine-ui[1571]
— XMMS[1572] - odtwarzacz multimedialny z licznymi wtyczkami
— cplay[1573] - odtwarzacz audio
— ecasound2.2[1574] - odtwarzacz audio
— mpd[1575] - odtwarzacz audio
— - odtwarzacz audio
— playmp3list[1576] - odtwarzacz audio
— splay[1577] - odtwarzacz audio
— amarok[1578] - odtwarzacz audio
— rhythmbox[1579] - odtwarzacz audio
— banshee[1580] - odtwarzacz audio

- noatun[1581] - odtwarzacz audio
- listen[1582] - odtwarzacz audio
- BMPx[1583] - odtwarzacz audio
- gwenview[1584] - przeglądarka graficzna (warto zainteresować się także wtyczkami kipi-plugins[1585] oraz programami pomocniczymi takimi jak: exiv2[1586])
- imgseek[1587] - przeglądarka graficzna
- showimg[1588] - przeglądarka graficzna
- fbi[1589] - przeglądarka graficzna wykorzystująca framebufer
- kview[1590] - przeglądarka plików graficznych dla KDE
- ksvg[1591] - przeglądarka SVG dla KDE
- djview[1592] - przeglądarka plików DjVu
- kdvi[1593] - przeglądarka DVI dla KDE
- kghostview[1594] - przeglądarka PS dla KDE
- kuickshow[1595] - przeglądarka plików graficznych dla KDE
- ogle-mmx[1596] - odtwarzacz wideo
- freevo[1597] - odtwarzacz wideo
- gphoton[1598] - odtwarzacz wideo
- SX[1599] - odtwarzacz wideo
- SMPlayer[1600] - odtwarzacz wideo

Programy do obsługi radia i telewizji

- tvtime[1601] - program do oglądania telewizji
- xawtv[1602] - program do oglądania telewizji (inne programy z tego zestawu pozwolą na słuchanie radia, czytanie telegazety, ...)
- alevt[1603] - softwarowy dekoder i przeglądarka teletekstu
- MythTV[1604] - kombajn do obsługi tunera)
- xdtv[1605] - - narzedzie do rejestracji obrazu z kart TV
- xawdecode[1606] (XdTV) -
- fftv[1607] -
- gradio[1608] - graficzny interfejs do sterowania tunerem radiowym
- GQradio[1609] - graficzny interfejs do sterowania tunerem radiowym
- fmtools[1610] - tekstowy interfejs do słuchania radia

Cała reszta

Oprócz programów dających się zaliczyć do wspomnianych grup istnieje bardzo wiele oprogramowania specjalistycznego, żeby wspomnieć choćby edytory schematów elektronicznych takie jak gEDA-graf, oprogramowanie do symulacji i obliczeń naukowych, ... No i w końcu gry, ale to już temat na osobny artykuł ...

- kgpg[1611] - narzędzie kryptograficzne (GPG) dla KDE
- kleopatra[1612] - narzędzie kryptograficzne (x509) dla KDE
- kompare[1613] - porównywacz plików / wyświetlacz plików diff dla KDE
- ksysguard[1614] - monitor systemu dla KDE (także w wariancie daemona - ksysguardd)
- ktimer[1615] - timer dla KDE
- shfs[1616] - montowanie systemu plików SFTP
- sshfs[1617] - montowanie systemu plików SFTP poprzez fuse
- Synergy[1618] - współdzielenie klawiatury i myszy poprzez sieć
- ajaxterm[1619] - terminal przez WWW
- anyterm[1620] - terminal przez WWW
- trickle[1621] - user-space bandwidth shaper
- aspell[1622] - system sprawdzania pisowni
- xwhois[1623] - klient whois działający w trybie graficznym
- kaddressbook[1624] - książka adresowa KDE
- kcalc[1625] - kalkulator KDE
- GNUPlot[1626] - program do tworzenia wykresów; więcej o gnuplocie
- plotutils[1627] -

- LabPlot[1628] - system analizy danych (wykresy, dopasowania) zgodny z Origin
- yacas[1629] - obliczenia algebraiczne
- maxima[1630] - obliczenia algebraiczne
- Octave[1631] - system obliczeniowy dobrze kompatybilny z MatLabem; więcej o GNU octave
- Scilab[1632] - pakiet naukowy stworzony do badań matematycznych
- StarDict[1633] - wielojęzyczny słownik
- recordMyDesktop[1634] - nagrywanie filmu z naszego ekranu
- Griffith[1635] - katalogowanie płyt z filmami
- partimage[1636] - tworzenie i przywracanie obrazów partycji
- clonezilla[1637] - sieciowe (z obsługa multicastu) tworzenie i przywracanie obrazów partycji
- Kvkbd[1638] - wirtualna klawiatura
- KLDraw[1639] - modelowanie LEGO
- KLDraw[1640] - modelowanie LEGO
- KLatexFormula[1641] - programik do edycji formuł LaTeX'owych
- Date Calculator[1642] - kalkulator dat
- Calenculator[1643] - kalkulator dat

– Nagrywanie CD i DVD

Dość istotną grupą programów są aplikacje umożliwiające zapis na dyskach optycznych (CD-R(W), DVD+-R(W), ...).

- K3b[1644] - graficzny program do obsługi nagrywania CD i DVD
- Cdrecord[1645] - tekstowy program do nagrywania CD (pakiet zawiera także `readcd` umożliwiający odczyt gołych danych z cd - podobnie jak `dd`, ale odrobinę bardziej inteligentnie)
- Cdrdao[1646] - tekstowy program do nagrywania CD
- wodim[1647] - zestaw programów umożliwiających zapis plików na CD/DVD ... wykorzystywany m.in. przez podawane powyżej nakładki graficzne
- UDFtool[1648] - zestaw programów umożliwiających zapis pakietowy na CD/DVD ... szczegóły poniżej
- DeVeDe[1649] - tworzenie DVD-video
- dvdisaster[1650] - testowanie i odczyt podniszczonych CD
- aptoncd[1651] - nagrywanie repozytorium na CD w oparciu o pakiety pobrane przez apt/aptitude
- dvd+rw-tools[1652] - narzedzi do obsługi zapisu płyt DVD+-RW
- udftools[1653] - zestaw narzędzi do obsługi (zapisu) UDF[1654]

Konfiguracja usług sieciowych

Na wstpępie zachęcam do zapoznania się z ogólnymi wiadomościami dotyczącymi usług sieciowych oraz samych sieci TCP/IP i ich konfiguraji.

Zawsze w przypadku instalowania i konfiguracji jakiejś usługi należy zapoznać się i ewentualnie poprawić plik konfiguracyjny (domyślne ustawienia najczęściej są OK, ale też prawie zawsze trzeba coś dostosować ...). Dalej przedstawię kilka uwag odnośnie konfiguracji niektórych z ważniejszych usług. Do najistotniejszych usług zaliczyłbym (w nawiasie przykładowe programy będące serwerem danej usługi): SSH/SCP/SFTP (sshd), DNS (bind), pocztę (exim), WWW (apache); warto też znaleźć miejsce na bramkę mail-www (squirrelmail), system bazodanowy (mysql) oraz webowy interfejs jego obsługi (phpmyadmin). Bardzo istotną kwestią jest także routing i podział łącza.

Niestety konfiguracja wielu (zwłaszcza tych młodszych) usług nie sprowadza się wyłącznie do przygotowania plików konfiguracyjnych - często zachodzi potrzeba przygotowania własnej kompilacji programu (z nałożonymi odpowiednimi łatkami) - ma to miejsce m.in. w przypadku ejabberd'a czy squirrelmail'a (szczegóły poniżej). Problemy nastręczają także próby utrzymania wspólnej bazy użytkowników dla wszystkich usług (konta shell, poczta, www, jabber, sip); niestety dodatkowo wiele z bardziej współczesnych programów ignoruje tradycyjny unixowy charakter kont (jedno konto w systemie, tradycyjne aliasy pocztowe, konfiguracja przechowywana w $HOME, ...) - zobacz poniżej informacje o zmaganiach z konfiguracją ejabberd'a i asterisk'a. W przypadku serwerów wielo-użytkownikowych dochodzi jeszcze problem przekazania maksymalnej kontroli nad konfiguracją swoich usług w ręce użytkowników (zobacz poniżej), ewentualnie jakiś panel przez nich zarządzany. Kolejnym problemem jest często obsługa vhostów i aliasingu domen (na serwerze konta kilku firm, i niektóre z nich chcą jeszcze aby wszystko było widoczne w obu ich domenach).

Generalnie w usługach serwerowych podobnie jak i w budowie sieci można wyróżnić kilka skal wielkości:

- siec wewnętrzna (firmowa, mieszkaniowa) - dystrybucja głównie pozioma, brak konieczności przekazywania dużej władzy w ręce poszczególnych użytkowników, rzadko kiedy potrzebne silne rozróżnienie usług w oparciu o nazwy domen (vhosty)
- siec budynkowa (w pojedynczym budynku mieszkalnym lub biurowym, ewentualnie niewielkiej grupie takich budynków) - dystrybucja najczęściej doprowadzająca sygnał do lokalu/biura, w dużej mierze pionowa, warto przekazywać dużą kontrolę nad usługami serwerowymi użytkownikom, na ogół konieczne rozbudowane vhosty
- siec osiedlowa - dystrybucja pozioma (pomiędzy budynkami) i pionowa (w ramach budynków), usług serwerowych brak lub ograniczony ich zakres lub sytuacja jak w sieci budynkowej
- duży ISP - dystrybucja pozioma na dużych obszarach, usług serwerowych brak lub ograniczony ich zakres, rzadko rozwinięte vhosty, częściej serwering dedykowany

Na wstępie przedstawię też przykład organizacji kont w sieci kamienicznej (jest to chyba najtrudniejszy organizacyjnie przypadek):

- grupy tworzone w oparciu o lokale - nazwa, oraz GID w jakiś sposób zgodna z numeracja lokali (np. dla lokalu nr. 23A - nazwa "23a", GUID 123+1000 = 1123, dla lokalu 23 "23" i 23+1000=1023)
- numeracja (klas) ip zgodna w jakiś sposób zgodna z numeracja grup (np. GID - 1000)
- konta umieszczane w grupach wg lokalu do którego przypisywany użytkownik
- nazwa konta odpowiada najkrótszemu aliasowi
- numeracja telefoniczna (podstawowe konto) tworzone w oparciu o UID posiadacza (prefix + UID - 1000)
- do konta 3 niezależne konfiguracje aliasów (mail, jabber, voip)
- warto tworzyć aliasy SIP generowane w oparciu o numer lokalu, przypisane do konta (podstawowego numeru) głównego użytkownika i dzwoniące na wszystkie telefony w lokalu (odpowiednie wywołanie `Macro(call_user|konto_glowne|telefony)` z moich przykładów konfiguracji asteriska)

Ogólnie na dużych systemach często dobrym pomysłem jest wyłączenie tworzenia indywidualnych grup dla każdego użytkownika i umieszczanie użytkowników w grupach w bardziej przemyślany sposób (patrz opcje `USERGROUPS` i `USERS_GID` w `/etc/adduser.conf`).

Poczta

Przy konfiguracji poczty warto zwrócić uwagę na możliwość zapewnienia jej filtrowania, przechowywanie jej w katalogach użytkowników oraz umieszczanie nagłówka informującego o adresie koperty - w przypadku exima jest to dość proste do uzyskania, zwłaszcza to ostatnie, które jest robione automnatycznie. Do filtrowania można wykorzystywać np. "procmail". Może on wraz z skanerami clamscan, spamassassin posłużyć do stworzenia systemu antywirusowego i antyspamowego (przykładowy plik konfiguracyjny filtra antywirusowego). Rozważyć należy również dodanie w głównym pliku ustawień poziomu logowania: `VERBOSE=no` i pliku logów `LOGFILE=/var/log/procmail.log`. Należy jednak zwrócić uwagę na to iż filtry uczące się zaczynają działać dopiero po przeanalizowaniu 200 pozytywnych i 200 negatywnych przykładów … (prezentowany skrypt uczy tylko negatywów), ustawienia te można zmienić w pliku konfiguracyjnym (więcej na jego temat w SpamAssassin - konfiguracja[1655]). Niekiedy (gdy filtr uczący się jest bardzo sprawny warto zwiększyć jego punktację (więcej o ustawianiu punktacji w SpamAssassin - testy[1656]).

Zamieszczam też dość bogato komentowany przykładowy plik konfiguracyjny exim'a, implementujący virusertable (przykład[1657] pliku z zdefiniowanymi standardowymi aliasami), obsługę procmaila (poprzez skrypt zapewniający poprawne logowanie do sysloga oraz obsługę globalnego pliku rc), dovecot'owych plików sieve, plików .forward, systemu vacation oraz tworzenie kopii odbieranej poczty po stronie serwera (dla poprawnego działania tego systemu należy zapewnić automatyczne kasowanie starych kopii - poprzez dodanie do cron'a odpowiedniego skryptu), a także ssl i autoryzację oraz graylisty (w oparciu o greylistd) i skanowanie antywirusowe i antyspamowe na etapie kopertowym. Dodatkowo polecam zmienić w `/etc/greylistd/config` czasy retryMin i retryMax na odpowiednio: 60 i 86400 (minuta i 24 godziny) oraz ustawić odpowiednią whitelistę (`/etc/greylistd/whitelist-hosts`) systemów mających problem z przejściem przez greylist lub po prostu godnych zaufania (przykład[1658]). Aby nasz system lepiej pokonywał greylisty na innych serwerach warto w `/etc/default/exim4` ustawić `QUEUEINTERVAL='3m'`.

Zamieszczam także zmodyfikowaną[1659] wersję tego pliku - służy ona do wysyłania kopii wychodzącej poczty na zadany adres i korzysta z zewnętrznego serwera SMTP wymagającego autoryzacji. Konfiguracja exima pozwala na odrzucanie wiadomości z informacją o nowym adresie (błąd SMTP 551 User not local), poprzez umieszczenie w pliku aliasów wpisu `::fail: 551 User not local; please try <inny@example.org>` niestety w przeciwieństwie do tego czego można by się spodziewać serwery pocztowe (w tym exim) nie rozróżniają tego od zwykłego błędu 550 i nie próbują przekazać poczty na wskazany adres, za to zwracają do adresata.

Przyjęte przeze mnie rozwiązanie z obsługą sieve[1660] ustawianego "zdalnie" (z wykorzystaniem jakiegoś klienta obsługującego ten mechanizm - np. squirrelmail z wtyczką avelsieve) polega na realizacji obsługi przez dovecot'a który w przypadku wykrycia takiego pliku (i braku plików o wyższym priorytecie) pełni funkcję LDA. Możliwe jest też korzystanie z filtrów sieve w samym eximie poprzez umieszczenie plików forward z odpowiednią pierwszą linią i zawartością będącą filtrem sieve.

Ważne jest zapewnienie dostępu do poczty przechowywanej na serwerze - służą do tego protokoły POP i IMAP. Polecam serwer dovecot, przy jego konfiguracji warto zwrócić uwagę na uruchomienie SSL (nieszyfrowany dostęp z zewnątrz można zabronić poprzez zamknięcie portów nieszyfrowanych, a zostawienie szyfrowanych na firewallu i ustawieniu `protocols = imap imaps pop3 pop3s` w konfiguracji dovecot) a także na ustawienie nasłuchiwania na ipv6 `listen = [::]` (oczywiście nasłuchuje wtedy także po IPv4). Zmaieszczam także przykładowy plik konfiguracyjny dla tego serwera - dovecot-locla.conf.

Dla poprawnego działania powyższych rozwiązań użytkownik Debian-exim powinien należeć do grup greylist i (w zależności od ustawień systemu) shadow, użytkownik dovecot do grup: mail i (w zależności od ustawień systemu) shadow, natomiast użytkownik clamav do grupy Debian-exim.

Popularnym i istotnym elementem systemów pocztowych jest bramka umożliwiająca dostęp do poczty poprzez WWW - np. Squirrel Mail[1661]. W systemie tym dla poprawnego kodowania wiadomości istotne jest posiadanie zainstalowanych odpowiednich locale. Większość takich systemów posiada dość rozbudowane opcje (konfiguracja tożsamości, książki adresowej, wersje robocze, ...), które mogą być powiększone przy pomocy dodatkowych modułów/patch'y. W przypadku wspomnianego już Squirrel Mail takie dodatki znaleźć możemy na http://www.squirrelmail.org/plugins.php[1662], na szczególną uwagę zasługują: View As HTML[1663] (jest nawet w paczce deb), Avelsieve - Sieve Mail Filters[1664] (jest nawet w paczce deb), Advanced Settings[1665] (np. wyświetlanie folderów specjalnych na początku, domyślne używanie priorytetów wiadomości), S/MIME verification[1666], Show Headers[1667]; a także: File Manager[1668], Calendar File Backend[1669], Notes[1670], To-Do Reminder[1671], NNTP webinterface[1672], Bookmarks[1673], Quick Save[1674], Autocomplete[1675], Custom From[1676], Message Flags & Icons[1677].

Aby konsolowy klient cone[1678] mógł łączyć się po SSL z serwerami mającymi błędny (np. samopodpisany) certyfikat w polu określającym adres serwera po jego nazwie lub numerze ip należy dodać: `/novalidate-cert`

EXIM4.CONF

```
# paczki dla serwera SMTP: exim4-daemon-heavy procmail spamassassin clamav-daemon greylistd

# exim -d+all -M 1JwyXH-0005nx-8z - przepuszczenie wskazanej wiadomosci z kolejki w trybie ⤸
debugowania (bez uwzgledniania czasow ponawiania)
# exim -bh host.OR.IP - testowanie regol dla poczty z zadanego adresu
# exim -brw adres     - testowanie "rewrite" dla podanego adreu
# exim -bt adres      - testowanie sciezki dostarczania dla podanego adresu

###
### KONFIGURACJA OGÓLNA
###

exim_path = /usr/sbin/exim4

# logi tylko przez syslog
log_file_path = syslog

# znacznik czasu dodawnay przez syslog
syslog_timestamp = false

# limit wielkości przesyłanego maila (gdy nie ustawiony to 50M)
message_size_limit = 1500M

daemon_smtp_ports = 25 : 465

# domeny dla których jesteśmy podstawowym MX
# @[] odpowiada za przyjmowanie "literałów domenowych" na wszystkie nasze IP
# kolejne nazwy domenowe w kolejnych liniach pliku,
# warto podać także nazwę podstawowa bierzącego systemu (@ może nienajlepiej działać)
```

```
# oraz nazwy skrucone hosta
domainlist local_domains = @ : @[] : localhost : localhost.localdomain : ⤸
/etc/mail/local_domains

# domeny dla których jesteśmy zapasowym MX
# będziemy próbować dostarczyć do serwera o najniższym MX
# kolejne nazwy domenowe w kolejnych liniach pliku
domainlist relay_to_domains = /etc/mail/relay_to_domains

# adresy IP na których słucha serwer
# local_interfaces = <; 127.0.0.1 ; 0.0.0.0 ; ::1 ; 2001:db8::1; [2001:db8::1]:8025

# podstawowa nazwa serwera pocztowego
# gdyby wczytanie z innych źródeł dawalo błędy ...
#primary_hostname = "moja.domena"
# instrukcje powyższą includujemy z innego pliku
.include_if_exists /etc/mail/primary_hostname

# hosty z których przyjmujemy pocztę
hostlist   relay_from_hosts = <; 127.0.0.1 ; ::1 ; /etc/mail/relay_from_hosts

# pozwalamy na przyjmowanie "literałów domenowych" czyli np. postamaster@[127.0.0.1]
allow_domain_literals

# ponawianie wysyłania
ignore_bounce_errors_after = 2d
timeout_frozen_after = 7d

# SSL
# generacja kluczy: openssl req -new -x509 -days 365 -nodes -out server.crt -newkey rsa:1024 -⤸
keyout server.key
tls_certificate = /etc/ssl/server.crt
tls_privatekey = /etc/ssl/server.key
tls_advertise_hosts = *
tls_on_connect_ports = 465

# autoryzacja dostępna tylko po SSL
auth_advertise_hosts = ${if eq{$tls_cipher}{}{}{*}}

# listy kontroli dostępu i filtrowania
# osobna kontrola po adresach dla poczty do nas i poczty na zewnatrz
acl_smtp_rcpt = ${if match_domain{$domain}{+local_domains} {acl_check_rcpt_in}{⤸
acl_check_rcpt_out}}
# działa to na zasadzie ${if forany{$local_domains}{eqi{$item}{$domain}} ...
acl_smtp_data = acl_check_data

# daemon antywirusowy wykorzystywany przez malware ACL
av_scanner = clamd:/var/run/clamav/clamd.ctl
#av_scanner = clamd:192.168.1.1 7575

# daemon antyspamowy wykorzystywany przez spam ACL - zostawiamy ustawienia domyslne
#spamd_address = localhost 783

###
### SEKCJE
###


### listy kontroli dostępu, akceptacji i filtrowania
begin acl
```

```
# kontrola poczty wychodzacej na zewnatrz w oparciu o relay_from_hosts i autoryzacje
acl_check_rcpt_out:
  # odrzucamy z niewlasiwymi znakami w adresie (kontrola poprawnosci adresu)
  deny message       = Restricted characters in address
   local_parts   = ^[./|] : ^.*[@%!] : ^.*/\\.\\./

  # wymagana weryfikacja adresu nadawcy
  require verify        = sender

  # przyjmujemy gdy poczta nie donas i pochodzi od hosta z relay_from_hosts (w to tez sie
  wlicza host lokalny)
  accept hosts         = +relay_from_hosts
   control       = submission

  # przyjmujemy gdy poczta nie donas i pochodzi od zautoryzowanego użytkownika
  accept authenticated = *
   control       = submission


# kontrola przychodzacej do nas w oparciu o local_domains i greylisting
#  + ustawienie polityki filtrowania dela acl_data
acl_check_rcpt_in:
  # wymagamy aby adresat byl u nas
  #  - inni adresaci sa odrzucani (na tym etapie nie rowazamy juz listow wysylanych przez
  nasz SMTP do innych
  #    a tylko listy do nas, do obcych jezeli uprawnione byly juz zaakceptowane wczesniej)
  #  - dziala po pojedynczych adresatach kopertowych
  require message = relay not permitted
   domains = +local_domains : +relay_to_domains


  # odrzucamy z niewlasiwymi znakami w adresie (kontrola poprawnosci adresu)
  deny message       = Restricted characters in address
   domains       = +local_domains
   local_parts   = ^[.] : ^.*[@%!/|]


  # podzial na grupy adresow wzgledem polityki filtrowania - ustawienia
  warn set acl_m_addr_mode = yes
  warn
   local_parts = postmaster : abuse : /etc/exim4/users_no_filter
   set acl_m_addr_mode = no
  warn
   !condition = ${if def:acl_m_data_mode}
   set acl_m_data_mode = $acl_m_addr_mode

  # czasowe odrzucenie gdy adres w innej polityce niz obecna
  #  dotyczy tylko drugich i kolejnych RCPT TO: oraz tylko adresow lokalnych
  defer message = Deferred - please try this recipient again later
   !condition = ${if eq{$received_protocol}{asmtp}}
   !condition = ${if eq{$acl_m_addr_mode}{$acl_m_data_mode}}


  # przyjmujemy na postmaster@_any_local_domain_ chyba ze w skutek poprzedniego RCPT TO
  zostalo wlaczone filtrowanie
  accept local_parts   = postmaster
   !condition    = ${acl_m_data_mode}


  # wymagana weryfikacja adresu nadawcy
```

```
  require verify        = sender

  # wymagana weryfikacja adresu odbiorcy
  require verify       = recipient

  # weryfikacja SPF (niedostępna w standarowej paczce)
  #warn
  # spf = pass:fail:softfail:none:neutral:err_perm:err_temp
  # add_header = X-SPF: $spf_header_comment

  # czasowe odrzucanie w oparciu o greylist, patrz: http://projects.puremagic.⤸
  com/greylisting/ (paczka Debiana: greylistd)
  defer
   message          = $sender_host_address is not yet authorized to deliver \
                      mail from <$sender_address> to <$local_part@$domain>. \
                      Greylisting in action, please try later.
   log_message      = greylisted.
   # !condition       = ${if and{ { def:spf_result }{ eq{$spf_result}{pass} } } }
   !local_parts     = ${if exists {/etc/exim4/users_no_graylist}{/etc/exim4/users_no_graylist}⤸
   {}}
   !local_parts     = postmaster
   !senders         = :
   !hosts           = : +relay_from_hosts : \
                      ${if exists {/etc/greylistd/whitelist-hosts}\
                           {/etc/greylistd/whitelist-hosts}{}} : \
                      ${if exists {/var/lib/greylistd/whitelist-hosts}\
                           {/var/lib/greylistd/whitelist-hosts}{}}
   !authenticated   = *
   domains          = +local_domains : +relay_to_domains
   verify           = recipient/callout=20s,use_sender,defer_ok
   condition        = ${readsocket{/var/run/greylistd/socket}\
                        {--grey $sender_host_address $sender_address $local_part@$domain}\
                        {5s}{}{false} }

  # akceptujemy całą resztę która dotarła do tego miejsca
  accept

# weryfikacja treści - antyspamy i antywirusy ...
acl_check_data:
  # akceptujemy gdy wszystkie adresy nie do nas
  accept !condition = ${if def:acl_m_data_mode}

  # akceptujemy gdy adresy z wylaczonym filtrowaniem
  accept !condition = $acl_m_data_mode

  # oznaczanie poczty wynikiem z SpamAssassin
  warn
   condition = ${if < {$message_size}{80K}}
   spam = nobody:true
   add_header = X-Spam-Score: $spam_score ($spam_bar)
   add_header = X-Spam-Report: $spam_report

  # odrzucamy poczte majaca powyzej 10 pkt w tescie antyspamowym
  deny message = This message was classified as SPAM
   condition = ${if >{$spam_score_int}{100}}

  # odrzucamy poczte z wirusami, patrz: http://www.exim.org/exim-html-4.⤸
  67/doc/html/spec_html/ch41.html
  # aby to dzialalo clamav powinien nalezec do grupy Debian-exim (usermod -G Debian-exim ⤸
  clamav)
```

```
  deny message = This message contains malware ($malware_name)
   condition = ${if < {$message_size}{200K}}
   demime = *
   malware = */defer_ok

  # akceptujemy całą resztę która dotarła do tego miejsca
  accept


### reguły kierowania do agentów dostarczania
begin routers

# dostarczanie do innych hostów - po domenie
dnslookup:
  driver = dnslookup
  check_srv = smtp

  domains = ! +local_domains
  ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8

  transport = remote_smtp

# dostarczanie do innych hostów - po ipliteral, gdy nie dało się po domenie
domain_literal:
  driver = ipliteral

  domains = ! +local_domains
  ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8

  transport = remote_smtp

# prymitywny odpowiednik driver'a iplookup ... służy do sprawdzania /etc/hosts
iplookup:
  driver = manualroute
  route_list = * $0 byname

  domains = ! +local_domains
  ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8

  transport = remote_smtp

  no_more
  # dalej przechodzi tylko local_domains ...

# obsługa /etc/mail/virtusertable
# wbrew sugestii *_transport ... pozostały routing jest przetwarzany i obsługiwany (np. ⤸
procmail)
# format pliku
# user@domena : user1, user2@domena1, user3@%2, %1@domena2, ...
# w powyąszym przykładzie "%1@" zostanie zastąpione przez "user@" oraz "@%2" przez ⤸
"@domena"
# można podawać samo ^user@.* lub ^.*@domena (uwaga jest to wyrażenie regularne ⤸
rozpoczynane od ^)
#
# możemy także korzystać z dyrektyw :include: :defer: :fail: np:
# user@domena : :include:/etc/mail/lists/list1
# (dostarczenie do listy adresów wymienionej w pliku /etc/mail/lists/list1)
# ^@domena.* : :fail: Unknow User
# (w @domena nie ma więcej użytkowników (niż wymienieni wcześniej w pliku aliasów),
# w szczególności nie ma użytkowników kont shelowych)
```

```
#
# zamiast forwardów możnaby próbować użyć  "::fail: 551 User not local  please try <⤸
inny@example.org>"
# (czyli odrzuć z podaniem nowego adresu), ale większość serwerów nie spróbuje ⤸
wykożystać tego adresu
# warto zaznaczyc że jeżeli są one zaraz po kolumnie alisu to należy je odzielić ⤸
dodatkowym :
# (jak ma to miejsce w powyższym przykładzie)
#
# możliwe jest także stosowanie dostarczania bezpośredniego (poprzedzenie nazwy \\)
# (w zasadzie wydaje się że powinien starczać jeden \, ale praktyka pokazuje że trzeba ⤸
dawać dwa ...)
# oraz dostarczania do programów (|sciezka_lub_nazwa )
virtusertable_full_domain:
  driver = redirect
  allow_fail
  allow_defer
  data = ${expand:${sg\
   {${sg\
    {${lookup{$local_part@$domain}nwildlsearch{/etc/mail/virtusertable}}}\
    {%1@}\
    {$local_part@}\
   }}\
   {@%2}\
   {@$domain}\
  }}
  file_transport = address_file
  pipe_transport = address_pipe
  reply_transport = address_reply

# kopie poczty w /home/mail/kopie
mail_backup:
  driver = accept

  # wymaga lokalnego użytkownika
  domains = +local_domains
  check_local_user

  # ignorowanie przy weryfikacji adresu oraz przy komendzie EXPN
  no_verify
  no_expn

  # przekazujemy kopię wiadomości
  unseen
  user = ${local_part}

  transport = write_mail_backup

# odpowiedzi automatyczne w oparciu o ~/exim_vacation.msg
user_vacation:
  driver = accept

  # wymaga lokalnego użytkownika i pliku ~/exim_vacation.msg
  check_local_user
  require_files = $home/exim_vacation.msg

  # nie odpisujemy na błędy bądź listy dyskusyjne
  condition = "${if or {{match {$h_precedence:} {(?i)junk|bulk|list}} {eq {$sender_address} {}⤸
  }} {no} {yes}}"
  senders = " ! ^.*-request@.*:\
```

```
    ! ^.*@list*.*:\
    ! ^owner-.*@.*:\
    ! ^postmaster@.*:\
    ! ^listmaster@.*:\
    ! ^mailer-daemon@.*\
    ! ^root@.*"

  # ignorowanie przy weryfikacji adresu oraz przy komendzie EXPN
  no_verify
  no_expn

  # przekazujemy kopię wiadomości
  unseen
  user = ${local_part}

  transport = vacation_reply

# obsługa plików ~/.forward
userforward:
  driver = redirect

  # wymaga lokalnego użytkownika i pliku $home/.forward
  # plik ten moze zawierac tradycyjne przekierowania, filtry sieve lub filtry exim
  # patrz: http://www.exim.org/exim-html-4.30/doc/html/filter_toc.html
  domains = +local_domains
  check_local_user
  require_files = $local_part:$home/.forward
  file = $home/.forward

  # ignorowanie przy weryfikacji adresu oraz przy komendzie EXPN
  no_verify
  no_expn

  check_ancestor
  allow_filter

  file_transport = address_file
  directory_transport = address_directory
  pipe_transport = address_pipe
  reply_transport = address_reply

# obsługa dovecotowych plików sieve - ~/.dovecot.sieve
# robimy to z użyciem dovecot'a ...
# możliwa jest także obsługa sieve z użyciem avelsieve - patrz http://dfoerster.⤸
de/opensource/exim-sieve-howto.html
dovecot_sieve:
  driver = accept

  # wymaga lokalnego użytkownika i pliku $home/.dovecot.sieve
  domains = +local_domains
  check_local_user
  require_files = $local_part:$home/.dovecot.sieve

  # ignorowanie przy weryfikacji adresu oraz przy komendzie EXPN
  no_verify
  no_expn

  transport = dovecot_pipe

# obsługa plików ~/.procmailrc
```

```
procmail:
  driver = accept

  # wymaga lokalnego użytkownika, obecnosci /usr/bin/procmail
  #  i pliku $home/.procmailrc lub /etc/procmailrc
  domains = +local_domains
  check_local_user
  require_files = ${local_part}:\
   ${if exists{/etc/procmailrc}\
   {/etc/procmailrc}{${home}/.procmailrc}}:\
   +/usr/bin/procmail

  # ignorowanie przy weryfikacji adresu oraz przy komendzie EXPN
  no_verify
  no_expn

  transport = procmail_pipe

# dostarczanie lokalne do ~/mail/INBOX
localuser:
  driver = accept

  # wymaga lokalnego użytkownika
  domains = +local_domains
  check_local_user

  transport = local_delivery
  cannot_route_message = Unknown user


### konfiguracja agentów dostarczania
begin transports

# przekazywanie do innych hostów w sieci
remote_smtp:
  driver = smtp

# dostarczanie lokalne do MAILDIR ~/MAIL/INBOX
local_delivery:
  driver = appendfile
  maildir_format
  directory = $home/MAIL/INBOX
  create_directory
  directory_mode = 0700
  delivery_date_add
  envelope_to_add
  return_path_add
  # znaczenie powyższych opcji zobacz w http://www.exim.org/exim-html-⤸
  current/doc/html/spec_html/ch44.html

# dostarczanie przez procmail
procmail_pipe:
  driver = pipe
  path = "/bin:/usr/bin:/usr/local/bin"
  log_output = yes
  command  = "/etc/exim4/procmail.sh /etc/mail/procmail_users/$local_part"
  #command = "procmail -t ORGMAIL=$home/MAIL/INBOX/ DEFAULT=$home/MAIL/INBOX/"
  return_path_add
  delivery_date_add
  envelope_to_add
```

```
# dostarczanie przez dovecot
dovecot_pipe:
  driver = pipe
  log_output = yes
  command = /usr/lib/dovecot/deliver
  message_prefix =
  message_suffix =
  return_path_add
  delivery_date_add
  envelope_to_add

# auto-responder
vacation_reply:
  driver = autoreply
  file = $home/exim_vacation.msg
  file_expand
  from = System Automatycznej Odpowiedzi <$original_local_part@$original_domain>
  log = $home/.exim_vacation.log
  once = $home/.exim_vacation.db
  once_repeat = 7d
  subject = ${if def:h_Subject: {Re: ${quote:${escape:${length_50:$h_Subject:}}} (autoreply)} ⤸
  {Informacja} }
  text = "!!! WIADOMOŚĆ WYGENEROWANA AUTOMATYCZNIE !!!\nPoniżej informacja od użytkownika ⤸
  <$original_local_part@$original_domain>\n\n"
  to = "$sender_address"

# agenty techniczne (m.in. na potrzeby ~/.forward)
address_reply:
  driver = autoreply
address_pipe:
  driver = pipe
  return_output
address_file:
  driver = appendfile
  delivery_date_add
  envelope_to_add
  return_path_add
address_directory:
  driver = appendfile
  delivery_date_add
  envelope_to_add
  return_path_add
  check_string = ""
  escape_string = ""
  maildir_format

# kopie zapasowe poczty
write_mail_backup:
  driver = appendfile
  user = mail
  group = mail
  mode = 0660
  mode_fail_narrower = false
  home_directory=/home/mail/kopie/
  file = /home/mail/kopie/$local_part
  delivery_date_add
  envelope_to_add
  return_path_add
```

```
### ponawianie wysyłania
begin retry
*   *   F,6m,2m; F,15m,5m; F,2h,15m; G,16h,1h,1.5; F,4d,6h


### przetwarzanie nagłówków
begin rewrite
# zamiana adresu nadawcy wg danych z pliku /etc/mail/from-addresses-map o formacie:
# stary@adres.pocztowy: nowy@adres.pocztowy
*@*    ${lookup{$1@$2}lsearch{/etc/mail/from-addresses-map}{$value}fail} frFs

### autoryzacja
begin authenticators
plain:
  driver = plaintext
  public_name = PLAIN
  server_prompts = :
  server_condition = "${if pam{$2:$3}{1}{0}}"
  server_set_id = $2

login:
  driver = plaintext
  public_name = LOGIN
  server_prompts = "Username:: : Password::"
  server_condition = "${if pam{$1:$2}{1}{0}}"
  server_set_id = $1

# UWAGA: autoryzacja przez PAM wymaga paczki exim4-daemon-heavy
#        oraz dodania użytkownika Debian-exim do grupy shadow
# konfiguracja PAMu w /etc/pam.d/exim4 powinna być następująca:
# #%PAM-1.0
#
# @include common-auth
# @include common-account
# @include common-session
```

EXIM4-SPY.CONF

```
###
### KONFIGURACJA OGÓLNA
###

# w tej konfiguracji serwer sporządza i przesyła na podany adres kopię wysyłanego listu
# wszystkie listy przesyłane są przez zewnętrzny SMTP wymagający autoryzacji
# (my nie jesteśmy serwerem pocztowym, nie mamy wpisów MX)

exim_path = /usr/sbin/exim4

# domeny dla których jesteśmy podstawowym MX
# @[] odpowiada za przyjmowanie "literałów domenowych" na wszystkie nasze IP
# kolejne nazwy domenowe w kolejnych liniach pliku,
# warto podać także nazwę podstawowa bierzącego systemu (@ może nienajlepiej działać)
# oraz nazwy skrucone hosta
domainlist local_domains = @ : @[] : localhost : localhost.localdomain

# domeny dla których jesteśmy zapasowym MX
# będziemy próbować dostarczyć do serwera o najniższym MX
# kolejne nazwy domenowe w kolejnych liniach pliku
```

```
#domainlist relay_to_domains = /etc/mail/relay_to_domains

# podstawowa nazwa serwera pocztowego
# gdyby wczytanie z innych źródeł dawalo błędy ...
primary_hostname = example.com

# hosty z których przyjmujemy pocztę
hostlist   relay_from_hosts = <; 127.0.0.1 ; ::1 ; /etc/mail/relay_from_hosts

# pozwalamy na przyjmowanie "literałów domenowych" czyli np. postamaster@[127.0.0.1]
allow_domain_literals

# ponawianie wysyłania
ignore_bounce_errors_after = 2d
timeout_frozen_after = 7d

# listy kontroli dostępu, akceptacji i filtrowania
acl_smtp_rcpt = acl_check_rcpt
acl_smtp_data = acl_check_data


###
### SEKCJE
###


### listy kontroli dostępu, akceptacji i filtrowania
begin acl

# kontrola w oparciu o local_domains i relay_from_hosts
acl_check_rcpt:
  # przyjmujemy z loklanego SMTP
  accept  hosts = :

  # kontrola poprawnosci adresu ...
  deny message       = Restricted characters in address
   domains       = +local_domains
   local_parts   = ^[.] : ^.*[@%!/|]
  deny message       = Restricted characters in address
   domains       = !+local_domains
   local_parts   = ^[./|] : ^.*[@%!] : ^.*/\\.\\./

  # przyjmujemy na postmaster@_any_local_domain_
  accept local_parts   = postmaster
   domains       = +local_domains

  # wymagana weryfikacja adresu nadawcy
  require verify        = sender

  # przyjmujemy gdy poczta od hosta z relay_from_hosts
  accept hosts         = +relay_from_hosts
   control       = submission

  # przyjmujemy gdy poczta od zautoryzowanego użytkownika
  accept authenticated = *
   control       = submission

  # gdy kilka to odrzucamy nie do nas gdy nieuprawniony
  require message = relay not permitted
   domains = +local_domains : +relay_to_domains
```

```
  # wymagana weryfikacja adresu odbiorcy
  require verify = recipient

  # całą resztę która do tąd dotarła akceptujemy
  accept

# weryfikacja treści - tu można wstawiać antyspamy i antywirusy ...
acl_check_data:
  accept


### reguły kierowania do agentów dostarczania
begin routers

# kopie wszystkich przesyłanych maili wysylamy na adres tajny.wspolpracownik@urzad.↩
bezpieczenstwa.gov.pl
outcoming_copy:
  driver = redirect
  domains = ! +local_domains
  data = "tajny.wspolpracownik@urzad.bezpieczenstwa.gov.pl"
  file_transport = address_file
  pipe_transport = address_pipe
  # ponizsza linijka jest BARDZO WAZNA
  unseen
# wariant z zapisem lokalnym
out_backup:
  driver = accept
  # gdy chcemy archiwizowac wewnetrzna wywalic ponizsza linijke
  # oraz " : 127.0.0.0/8" z ignore_target_hosts
  domains = ! +local_domains
  transport = out_backup_write
  ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8
  # ponizsza linijka jest BARDZO WAZNA
  unseen
# przekazywanie do innych hostów w sieci poprzez zdalny SMTP
dnslookup:
  driver = manualroute
  domains = ! +local_domains
  transport = remote_smtp
  route_list = * smtp.u.naszego.isp byname
  host_find_failed = defer
  same_domain_copy_routing = yes
  ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8
  no_more

# systemowe aliasy
system_aliases:
  driver = redirect
  domains = +local_domains
  data = ${lookup{$local_part}lsearch{/etc/aliases}}

# mozemy także zdefiniowac alias bezpośrednio w konfigu exima a nie w jakimś pliku (typu ↩
/etc/aliases)
#local_alias:
# driver = redirect
# domains = +local_domains
# data = ${if eq {$local_part}{root} {root@mail-adm.vls.icm.edu.pl}{$local_part@$domain}}

# także wszystkie maile lokalne (do root itp) wysylamy na tajny.wspolpracownik@urzad.↩
```

```
bezpieczenstwa.gov.pl
localuser:
  driver = redirect

  # wymaga lokalnego użytkownika
  domains = +local_domains
  check_local_user

  data = "tajny.wspolpracownik@urzad.bezpieczenstwa.gov.pl"
  file_transport = address_file
  pipe_transport = address_pipe

  cannot_route_message = Unknown user


### konfiguracja agentów dostarczania
begin transports

# nasz zdalny SMTP wymaga autoryzacji
remote_smtp:
  driver = smtp
  hosts_require_auth = smtp.u.naszego.isp

# zapis lokalnej kopii poczty wychodzącej
out_backup_write:
  driver = appendfile
  user = mail
  group = mail
  file = /var/backup/outgoing/$return_path
  delivery_date_add
  envelope_to_add
  return_path_add

address_reply:
  driver = autoreply
address_pipe:
  driver = pipe
  return_output
address_file:
  driver = appendfile
  delivery_date_add
  envelope_to_add
  return_path_add

### ponawianie wysyłania
begin retry
*   *   F,2h,15m; G,16h,1h,1.5; F,4d,6h


### przetwarzanie nagłówków
begin rewrite


### autoryzacja
begin authenticators
# konfiguracja autoryzacji do zdalnego smtp
plain:
  driver = plaintext
  public_name = PLAIN
```

```
  client_send = ^USERNAME^PASSWORD
```

DOVECOT-LOCAL.CONF

```
# słuchamy na wszystkich interfejsach w IPv6 i IPv4
# niestety w tej wersji jest wlaczony IPV6_V6ONLY, wiec robimy to osobno :-(
listen = *, [::]

# logi do sysloga
log_path = syslog
syslog_facility = mail

# grupa pocztowa
#mail_access_groups = mail

# certyfikaty SSL serwera
ssl_cert = </etc/ssl/server.crt
ssl_key = </etc/ssl/server.key

# automatycznie tworzymy brakujace skrzynki
lda_mailbox_autocreate = yes

# nagłówek zawierający odbiorcę kopertowego
lda_original_recipient_header = "Envelope-to"

# adres nadawcy dla zwrotek itp
postmaster_address = postmaster@example.org

# obslugiwane protokoly: IMAP (takze imaps) oraz SIEVE
protocols = "imap sieve"

# w sieve sluchamy i na nowym i na starym standardzie
service managesieve-login {
  inet_listener sieve {
    port = 4190
  }

  inet_listener sieve_deprecated {
    port = 2000
  }
}

plugin {
  sieve = ~/.dovecot.sieve
  sieve_dir = ~/.sieve
}

protocol lda {
  mail_plugins = $mail_plugins sieve
}

# poczte trzymamy w $HOME/MAIL/ w formacie maildir, inbox jest w $HOME/MAIL/INVBOX
# pliki wewntrze dovecot'a trzymamy w $HOME/MAIL/.dovecot
mail_location = maildir:~/MAIL:INBOX=~/MAIL/INBOX:LAYOUT=fs:CONTROL=~/MAIL/.dovecot:⤸
INDEX=~/MAIL/.dovecot
```

ROTATE_MAIL_BACKUP.SH

```
#!/bin/bash

# wyciszenie skryptu ...
exec 1> /dev/null
exec 2> /dev/null

#e-mail
cd /home/mail/kopie/ ;
savelog -r older -m 640 -u mail -g mail -c 2 * ;

exit 0
```

PROCMAIL.SH

```
#!/bin/bash

procmail -t RCFILE="$1" /etc/exim4/procmail.rc 2>&1 | /usr/bin/logger -t "procmail[$PPID]" -p ⤸
mail.debug
```

PROCMAIL.RC

```
ORGMAIL=$HOME/MAIL/INBOX/
DEFAULT=$HOME/MAIL/INBOX/
UMASK=033
VERBOSE=yes

:0
* ^Content-Type: *text/plain
{
:0 fbw
* ^Content-Transfer-Encoding: *base64
| base64 -d
# base64 is in package   coreutils

:0 Afhw
| formail -I "Content-Transfer-Encoding: 8bit"

:0 fbw
* ^Content-Transfer-Encoding: *quoted-printable
| mewdecode -d -q
# mewdecode is in package mew-bin

:0 Afhw
| formail -I "Content-Transfer-Encoding: 8bit"
}

INCLUDERC="$HOME/.procmailrc"
INCLUDERC="$RCFILE"
```

Programy wspierające - procmail, vacation, fetchmail i plik /.forward)

Plik .forward (jest to standardowa ścieżka do tego pliku) umieszczany w katalogu domowym pozwala na przekazywanie poczty na inny adres (adres w pełnej postaci (z @), nazwa lokalnego użytkownika, aby dostarczyć bezpośrednio do skrzynki (z pominięciem forward'ów) należy adres poprzedzić znakiem \), na kilka adresów (oddzielamy je przecinkiem) oraz do innych programów ("|polecenie").

Program procmail pozwala na sortowanie, filtrowanie, przesyłanie, automatyczne odpowiadanie na pocztę, pocztę do niego można przekazać przez plik .forward może też być automatycznie dostarczana przez systemowy MTA. Plikiem konfiguracyjnym tego programu zazwyczaj jest .procmailrc. Każda reguła zapisana w tym pliku składa się z kilku linii, zaczyna się od linii postaci: :0 [flagi] [:[plik-blokujący]], do ważniejszych flag należy zaliczyć A - wykonanie reguły zależne od warunku z poprzedniej reguły i c - wykonanie kopii wiadomości, po linii tej następują linie warunków (jeden warunek w linii zaczynającej się od *, opcjonalnie można pominąć linię z warunkiem) a po nich następuje linia zawierająca akcję (linia taka w jednej regule może być tylko jedna !); reguły możemy zagnieżdżać przy pomocy nawiasów klamrowych { } i \mintinline{txt}{} (po nawiasie otwierającym musi wystąpić spacja tabulator lub nowa linia). Zobacz także: man 1 procmail, man 5 procmailrc, man 5 procmailex

```
# przykładowy plik .procmailrc
# test zaczynający się od znaku # to komentarz i jest on ignorowany przez procmail
# gdy nie ma w naglowku pola "X-Envelope-To:" to dodaje pole i przekazuje do ponownie
↪  procmaila
# jako wartość pola (zmienna $dla) ustawiany jest pierwszy ($1) argument przekazany do
↪  procmaila'
# przy wywolaniu go z pliku jako lokalnego agenta dostarczania z pliku sendmail.cf -
↪  makro $h
#
# będzie on ustawiony na user1@domena gdy przy przekazywaniu poczty z tego adresu
# do rzeczywistego użytkownika - user (pliki aliases, virtusertable i .forward)
# podane będzie przekazanie do user+"user1@domena"
# (znaki "" służą do zabezpieczneia symbolu @ i będą w zmiennej wynikowej)
# zamiast adresu user1@domena można podawac dowolny inny tekst
dla=$1
:0
*!X-Envelope-To:
| formail -i "X-Envelope-To: "$dla |procmail
# pocztę adresowaną (opisywane wyżej pole X-Envelope-To:)
# do user1@domena1 umieszczam w skrzynce (pliku) mail/skrzynka1
:0
*X-Envelope-To: "user1@domena1"
mail/skrzynka1
# poczte mającą w temacie "wazne" przesyłam na swoje drugie konto
↪  (login@domena.drugiego.konta)
# i do programu vacation aby odpowiedzial na list
# jest to jedno z możliwych rozwiązań - najbardziej logiczne
:0
* ^Subject:.*wazne
{
        :0 c
        ! login@domena.drugiego.konta
        :0
        |vacation
}
# powiadomienia o wiadomosciach IM
:0 :
*From:.*xmpp-system@n17.waw.pl
mail/jabber
# wysylanie powiadomien poprzez IM o mailach które dotarły do teego miejsca
INCLUDERC=/path-to/notify_to_xmpp
# na poczte od konto@jakas.domena odrazu odpowiadam
```

```
# (wysylajac zawartosc pliku odpowiedz.txt) i jej nie zapisuje
:0
* ^From:.*konto@jakas.domena
| (formail -r ; cat odpowiedz.txt) | $SENDMAIL -oi -t
```

Innymi przykładami wykorzystania procmaila mogą być:

- konwersja na kodowanie 8bitowe (z base64 i quoted-printable) - konwert_to_8bit[1679]
- filtrowanie wirusów - wirus_check, wirus.sh
- filtrowanie spamu - spam_check[1680], spam_mv[1681], naukę można zrealizować dodając do cron'a `sa-learn --mbox --spam ~/mail/spam_niewykryty; :> ~/mail/spam_niewykryty`
- powiadamianie przez XMPP notify_to_xmpp, notify_to_xmpp_chat[1682]

Program `vacation` pozwala na proste automatyczne generowanie odpowiedzi, pocztę do programu tego przekazujemy z pliku `.forward` (warto pamiętać o bezpośrednim przekazaniu listu do swojej skrzynki). Wiadomość zwracana przez ten program przechowywana jest w pliku `.vacation.msg`, plik ten rozpoczynać się może od pól nagłówków (takich jak `Subject:` (zawierającego tytuł), `From:` (zawierającego adres nadawcy, przy braku tego pola zostanie ono wygenerowane automatycznie), `Reply-To:` (informującego o tym gdzie należy wysłać odpowiedź), w treści tego pliku można także skorzystać ze zmiennej `$SUBJECT` wstawiającej tytuł wiadomości, na którą generowana jest odpowiedź. Przydatnymi opcjami programu są `-j` nakazująca odpowiadać m.in. na przeforwardowane wiadomości (nie sprawdza pola To: ani Cc:) oraz opcja `-t[N]` nakazująca wysyłanie tylko jednego powiadomienia na dany adres w ciągu N dni (aby wyczyścić bazę tych adresów należy wywołać `vacation -I`). Uwaga na wywołanie samego `vacation` otwiera ono edycję pliku `.vacation.msg` i zmienia plik `.forward`, zapisując starą wersję jako `.forward.old`. Zobacz także: `man 1 vacation`

Program `fetchmail` pozwala na pobieranie (zarówno automatyczne jak i ręczne) poczty z zdalnych serwerów i umieszcza je w skrzynce systemowej (działa tak jak forward z komputera z którego pobiera pocztę). Plikiem konfiguracyjnym tego programu zazwyczaj jest `.fetchmailrc`, plik ten zawiera konfiguracje kont pocztowych wraz z nazwami użytkowników i hasłami (one są opcjonalne przy uruchamianiu manualnym, ale ze względu na to prawa do pliku muszą przysługiwać tylko właścicielowi) podstawowy wpis składa się z pozycji: `poll "nazwa serwera" username "użytkownik" password "hasło"`, dodanie opcji `keep` powoduje pozostawienie wiadomości na serwerze. Opcja `-d 600 -N` pozwala na uruchomienie programu w trybie pracy ciągłej gdzie sprawdzanie poczty odbywa się co 600 sekund, opcja `-f` pozwala określić alternatywny plik konfiguracyjny i przydaje się, gdy trzeba wykonać jakieś jednorazowe sprawdzenie, a fetchmail działa w trybie pracy ciągłej. Jeżeli jest to możliwe, zawsze należy używać forward'u zamiast fetchmail'a. Jeżeli zachodzi potrzeba rzadszego uruchamiania fetchmail'a (np. kilka razy na dobę), to warto rozważyć skorzystanie z systemowego mechanizmu corn, pozwalającego na uruchamianie programów o zadanej porze, w tym celu należy uruchomić `crontab nazwa-pliku` aby utworzyć wpis z podanego pliku, każda linia pliku powinna wyglądać następująco: `minuta godzina dzień miesiąc dzień tygodnia komenda` (przy czym dzień tygodnia - liczba od 0 (niedziela) do 6 (sobota) a także każdy z parametrów można zastąpić gwiazdką co oznacza że dla każdej wartości tego parametru będzie następowało uruchomienie. Zobacz także: `man 1 fetchmail`

WIRUS.SH

```
#!/bin/bash

file_name=`mktemp`
cat > $file_name
wynik=`nice -n5 clamscan --stdout --recursive --mbox $file_name`
#echo "$wynik" > /tmp/wirusy
echo "$wynik"
rm -f $file_name
```

WIRUS_CHECK

```
# czyscimy po obcych antywirusach
:0 fhw
| formail -I "X-AntiVirus:"

# skanujemy wiadomosci (jezeli niezbyt duze)
:0 hbw
* < 200000
```

```
AV_REPORT=|/path-to/wirus.sh
#AV_REPORT=|clamscan --stdout -v --recursive --mbox -

# oznaczamy wiadomosci w oparciu o wynik skanowania
:0 fhw
* < 200000
| formail -I "X-AntiVirus:`echo "$AV_REPORT" | awk 'BEGIN { printf(" OK") } $NF=="OK" { ⤸
printf(" OK") }  $NF=="FOUND" { printf(" %s", $(NF-1)) }'`"

# zmiana tematu wiadomosci oznaczonych jako zawirusowane
:0 fhw
*^X-AntiVirus:
*!^X-AntiVirus:.*Oversized
*!^X-AntiVirus:( OK)+$
{
:0 hw
subject=|formail -xSubject:
:0 fhw
| formail -I "Subject: !!! UWAGA WIRUS !!! - $subject"
}
```

NOTIFY_TO_XMPP

```
# wyslanie powiadomienia na JABBER
:0 hw
subject=|mail2text header "`formail -xSubject:`"
:0 hw
from=|mail2text header "`formail -xFrom:`"
:0 h
*!From:.*xmpp-system@n17.waw.pl
jabber=|echo "Nowy e-mail: $subject (od: $from)." | smtp2xmpp --to "$LOGNAME@"`hostname -d`
# aby powiadomienia były wysyłane jako wiadomości a nie nagłówki nalezy do smtp2xmpp ⤸
dodać opcję --type ""
```

MAILMAN

Mailman jest jednym z najpopularniejszych serwerów obsługujących pocztowe listy dyskusyjne[1683]. Do zarządzania nim oprócz interfejsu WWW możemy kożystać także z narzędzi shellowych takich jak: `newlist`, `list_lists`, `config_list` (wyświetlenie konfiguracji listy w postaci tekstowej: `config_list -o - NAZWALISTY`, zmiana jednego parametru: `echo "archive_private = 1" > /tmp/aa; config_list -i /tmp/aa NAZWALISTY`). Warto także pamiętać że ustawienia domyślne z brane są z pliku `/usr/lib/mailman/Mailman/Defaults.py`,a nadpisywane są przez ustawienia w `/etc/mailman/mm_cfg.py`.

XMPP/Jabber

Ciekawą i interesującą usługą jest Instant Message, a konkretniej protokół XMPP[1684]. O protokole tym i jego bolączkach wieku dziecięcego wspomniałem już wcześniej.

Główna bolączką XMPP wydaje się być opóźnienie implementacji w stosunku co do standardów. W przypadku serwerów nie jest bardzo tragicznie (większość funkcjonalności jest dostępna w formie dodatkowych modułów lub łatek), natomiast w przypadku klientów braki są liczniejsze (tutaj jednak zawsze można sobie poradzić komunikując się z nieobsługiwaną w kliencie funcjonalnością przy pomocy konsoli XML).

Najciekawszym w chwili obecnej serwerem tej usługi wydaje się być ejabberd[1685]. Zamieszczam jego przykładową konfigurację - `ejabberd.cfg`.

Warto zainteresować się również łatkami do tego serwera zwiększającymi jego funkcjonalność mod_archi-

ve_odbc-release[1686], dodającym implementację Message Archiving (XEP-0136), jest także jego nowsza wersja pod nazwą mod_archive2[1687] (wersja odbc używana z PSQL 8.4.12 i erland R14 może wymagać załatania mod_archive_webview - spodziewa się tekstowej postaci czasu UTC, ale otrzymuje z bazy postać numeryczną - patch mojego autorstwa[1688]). Kolejnym ciekawym patchem jest mod_multicast[1689] implementujący wieloadresowanie Extended Stanza Addressing (XEP-0033) oraz patcjh implementujący XEP-0079: Advanced Message Processing[1690]. Zainteresować się można również patch'ami umożliwiającymi: dostęp do MUC przez IRC[1691], zapis informacji użytkowników do HTML[1692], logowanie pakietów jako XML[1693], filtrowanie dostępu[1694]. Ciekawe wydają się także moduły do współpracy z serwerem VoIP Asterisk - mod_asterisk[1695] i mod_client_asterisk[1696].

Wiecej patchy znaleźć można na: ejabberd.im[1697]. Wiele z opisywanych łatek jest do pobrania poprzez: `svn co https://svn.process-one.net/ejabberd-modules` i budowane są one niezależnie od samego ejabberda. Jednak patche mojega autorstwa związane z aliasami JID ("ejabberd_log_offline_message.diff" i "ejabberd_alias_forward.diff") wymagają pobrania źródeł ejabberda (`apt-get source ejabberd`) zapatchowania go (`cd ejabberd-*/src; patch -p0 < ../../ejabberd_*.diff`) i zbudowania (`./configure && make` lub `cd ..; dpkg-buildpackage -rfakeroot`).

Polecam także mój system powiadomień XMPP/SMTP zawierający m.in. łatką "ejabberd_log_offline_message.diff" powodującą zapisywanie wiadomości trybu offline także do plików tekstowych, które wykorzystuje w moim systemie do przesyłania powiadomień o nowych IM na e-mail. Zachęcam także do zapoznania się z zestawem moich patch'y do serwera ejabberd implementujących JID aliases.

Innymi wartymi zainteresowania uzupełnieniami serwera xmpp są z pewnością: transport xmpp2xmpp[1698] (ułatwia on między innymi bycie on-line na kilku kontach równocześnie, korzystanie z adresu na innym serwerze i usług takich jak archiwizacja na innym - JID aliasing) oraz system prezentacji i zarządzania historią rozmów - Jorge[1699].

Przy konfiguracji warto zwrócić uwagę na dostęp do MUC (polecam ustawienie umożliwiające udział każdemu a zakładanie tylko swoim - `{mod_muc,  [{access, muc}, {access_create, muc_create}, {access_admin, muc_admin}]  }`, polecam ustawić także: `{access, muc_admin, [{allow, admin}]}.` `{access, muc_create, [{allow, local}]}. {access, muc, [{allow, all}]}.`. Jeżeli nasz system kożysta z IPv6 należy uruchomić słuchania na tym protkole - wpisy inet6 w konfiguracji portów nasłuchiwania np.: `{5222, ejabberd_c2s, [inet6, {access, c2s}, ....]}`. W przypadku kożystania z autoryzacji przez PAM (wpis konfiguracyjny `{auth_method, pam}.`) konieczne może okazać się dodanie użytkownika ejabberd do grupy shadow, warto wtedy także wyłączyć możliwość rejestracji - `{access, register, [{deny, all}]}.`. Przykładowy konfig: ejabberd.cfg[1700].

Niekiedy (gdy za dużo napsujemy) mogą wystąpić problemy z startem ejabberd'a. Należy wtedy wyeksportować dane konfiguracyjne z jego bazy (`ejabberdctl dump`), zatrzymać ejabberda, usunąć `/var/lib/ejabberd` i po ponownym uruchomieniu ejabberda przywrócić z pliku tekstowego(`ejabberdctl load`). Można także bawić się z podmianą wybranych plików w tym katalogu, jest to przydatne tylko gdy nie możemy uruchomić ejabberd'a a nie posiadamy zrzutu - przenosimy wtedy ten katalog w inne miejsce, odpalamy ejabberda, zatrzymujemy go, pliki LATEST.LOG config.DCD local_config.DCD disco_publish.DAT z nowo utworzonego katalogu kopiujemy do starego, zastępujemy nim nowy i odpalamy ejabberda. Ta metoda w odróżnieniu od pierwszej regeneruje bazę jednorazowo, więc przy następnym starcie też będzie problem.

Jeszcze innym (i chyba najlepszym) rozwiązaniem tych problemów jest wywalenie zawartości `/var/lib/ejabberd`, odpalenie w celu uzyskania nowego konfigu, przywrócenie poprzedniej zawartości `/var/lib/ejabberd` i zrobienie `cd /path/do/zarhiwizowanego/_nowego_/var/lib/ejabberd/ && cp /tmp/ejabberd/config.DCD /tmp/ejabberd/local_config.DCD /tmp/ejabberd/disco_publish.DAT /var/lib/ejabberd/; chown ejabberd:ejabberd /var/lib/ejabberd/*; /etc/init.d/ejabberd start`

Interesującym rozwiązaniem jest także dobrze współpracujący z ejabberd webowy klient xmpp - jwchat. Korzysta on z mechanizmu HTTP Binding (lub HTTP Polling) oprócz włączenia tych mechanizmów w ejabberd (moduł mod_http_bind oraz odpowiednia konfiguracja ejabberd_http) i ustawienia dla nich proxy (m.in. po to aby puszczać to po https) należy szczególną uwagę zwrócić na to iż w url występuje - a w konfiguracji mod_http_bind występuje _.

Zobacz także: Jabber archive web reader[1701], wykaz rozszerzeń / protokołów wspieranych przez ejabberd[1702], wykaz rozszerzeń protokołu (dokumentów XEP)[1703], XEP-0209: Metacontacts, XEP-0153: vCard-Based Avatars.

Psi - kient XMPP

Program ten cechuje dośc nieprzyjemna cecha polegająca na niemożności zalogowania się bez podania hasła klucza GPG, gdy taki ustawiliśmy. Prezentuje patch mojego autorstwa zmieniający to zachowanie (pozwalający na zalogowanie bez podpisywania statusu) - patch[1704] (szczegóły[1705]).

Klient ten nie pozwala obecnie na korzystanie poprzez swój interfejs z archiwizacji wiadomości po stronie serwera (Message Archiving, JEP-0136). Mozna jednak usługę tą aktywować i używać z poziomu konsoli XML. Poniżej zamieszczam krótki przykład jak to robić:

```
<!-- SPRAWDZENIE -->
<iq type='get' id='save1' from='user@example.net'>
  <save xmlns='http://jabber.org/protocol/archive'/>
</iq>
<!-- WLACZENIE -->
<iq type='set' id='save2' from='user@example.net'>
  <save xmlns='http://jabber.org/protocol/archive'>
    <default save='true'/>
  </save>
</iq>
<!-- POBRANIE SPISU -->
<iq type='get' to='example.net'>
  <list xmlns='http://jabber.org/protocol/archive'
        maxitems='50'/>
</iq>
<!-- POBRANIE ZAPISU ROZMOWY -->
<iq type='get' to='example.net' id='down5'>
  <retrieve xmlns='http://jabber.org/protocol/archive'
            with='testowy@conference.example.net'
            start='2007-04-08T22:00:49Z'/>
</iq>
<!-- USUNIECIE WSZYSTKICH -->
<iq type='set' to='example.net'>
  <remove xmlns='http://jabber.org/protocol/archive'/>
</iq>
```

XML możemy także wykożystać do ustawienia innego statusu na jednym z transportów - np. na GG: `<presence type="available" to="gg.serwer.xmpp"><show>away</show><status>jeżeli możesz pisz na jabbera, JID: rrp@n17.waw.pl</status></presence>`

Powiadomienia XMPP/SMTP

Głównym celem prezentowanego przeze mnie systemu powiadomień XMPP/SMTP jest umożliwienie zapewnienia przepływu informacji o nowych wiadomościach pomiędzy tymi dwoma protokołami. System składa się z 4 skryptów:

— xmpp2email.sh - Wysyła powiadomienia o wiadomościach IM otrzymanych w trybie off-line na e-mail, wymaga przechowywania wiadomości off-line w bazie postgres. Jako argument przyjmuje nazwę użytkownika (nie jid) którego wiadomości mają zostać przetworzone. Do parsowania xml'a otrzymanego z bazy danych używany jest arkusz xls xmpp2email.xls. Program pełni rolę podobną do serwera powiadomień dla mikrokontrolera, przy czym tamten korzystał z bazy MySQL serwera jabberd2.
— smtp2xmpp.pl - Program wysyłający wiadomość IM z specyficznego adresu (podobny w działaniu do jabber.cpp, czy też sendxmpp), wiadomość wysyłana jest domyślnie jako headline, dzięki czemu nie powinna być przechowywana w trybie offline.
— mail-bot.pl - bot jabberowy udzielający informacji o nowej poczcie, do pobierania informacji o wiadomościach korzysta z "mail2text.sh"
— mail2text.sh - Pomocniczy program odpowiedzialny za konwersję kodowania mailowego znaków non-ascii z 7 lub 8 bitowego w jakiejś stronie kodowej na utf-8. Program oprócz konwersji nagłówków, posiada także funkcjonalność wyciągnięcia treści tekstowej z maila (również z maila typu mime zawierającego część text/plain) i dokonania jej konwersji. Przykład zastosowania razem z smtp2xmpp.pl podany jest w pliku procmailowym notify_to_xmpp.

Można także pobrać te programy w formie archiwum źródłowego paczki *.deb - xmpp-smtp.tgz[1706].

EMAIL2TEXT.SH

```
#!/bin/bash

# możliwe jest także wywołanie z jednym parametrem stanowiącym łańcuch tekstowy
# wykonywana jest wtedy konwersja kodowania base64/qp2 na 8 bit
# oraz odpowiedniego kodowania znaków narodowych do utf8
#
# wymagane paczki: mew-bin (dla mewdecode), konwert, gawk, procmail (dla formail)

convert_charset() {
 case "$1" in
  "ISO-8859-2" | "iso-8859-2")
   echo "konwert isolatin2-utf8"
   ;;
  "UTF-8" | "utf-8")
   echo "cat"
   ;;
  *)
   echo "cat"
   ;;
 esac
}

convert_encode() {
 case "$1" in
  "Q" | "q" | "quoted-printable" | " quoted-printable")
   echo "konwert qp-8bit"
   ;;
  "B" | "b" | "base64" | " base64")
   echo "mewdecode"
   ;;
  *)
   echo "cat"
   ;;
 esac
}

base64qp2utf8() {
 sed -e 's#=?\([^=]\)#\n=?\1#g' | sed -e 's#\([^?]*\)=?\([^?]*\)?\([qQbB]\)?\([^?]*\)?=\(.*\)⤸
 #\1\n\2 \3 =?@@@?= \4\n\5#g' |
   while read w1 w2 w3 w4; do
  if [ "$w3" = "=?@@@?=" ]; then
   echo "$w4" | `convert_encode "$w2"` | `convert_charset "$w1"` | sed -e 's#^ *##g' -e 's# ⤸
   *$##g'
  else
   echo "$w1 $w2 $w3 $w4" | sed -e 's#^ *##g' -e 's# *$##g'
  fi
 done | tr '\n' ' '
}

email2text() {
 # header
 DATE=`echo "$1" | formail -cx 'Date:'`
FROM=`echo "$1" | formail -cx 'From:' | base64qp2utf8`
SUBJ=`echo "$1" | formail -cx 'Subject:' | base64qp2utf8`
echo -e "DATE: $DATE\nFROM: $FROM\nSUBJECT: $SUBJ\nBODY:"

if [ "$2" != "nobody" ]; then
  # body
```

```
  CONT_ENCO=`echo "$1" | formail -cx 'Content-Transfer-Encoding:'`
  CONT_TYPE=`echo "$1" | formail -cx 'Content-Type:'`

  MIME_TYPE=`echo $CONT_TYPE | sed 's#\([^;]*\);.*$#\1#'`
  case "$MIME_TYPE" in
   "text/plain" | "")
    MIME_CHAR=`echo $CONT_TYPE | sed 's#[^;]*;.*charset="*\([^" ]*\)"*$#\1#'`
    echo "$1" | formail -I '' | `convert_encode "$CONT_ENCO"` | `convert_charset ⤸
    "$MIME_CHAR"`
    ;;
   "multipart/mixed" | "multipart/alternative")
    MIME_BOUN=`echo $CONT_TYPE | sed 's#[^;]*;.*boundary="*\([^"]*\)"*$#\1#'`
    SUBMESSAGE=`echo "$1" | formail -I '' |
     awk 'BEGIN {RS=".?.?'"$MIME_BOUN"'"} /Content-Type:.*text\/plain;/ {print $0; exit}'`

    CONT_ENCO=`echo "$SUBMESSAGE" | formail -cx 'Content-Transfer-Encoding:'`
    MIME_CHAR=`echo "$SUBMESSAGE" | formail -cx 'Content-Type:' | sed 's#[^;]*;.*charset="*\([⤸
    ^" ]*\)"*$#\1#'`
    echo "$SUBMESSAGE" | formail -I '' | `convert_encode "$CONT_ENCO"` | `convert_charset ⤸
    "$MIME_CHAR"`

    echo ""
    echo "[This is multipart message - get first text/plain section]"
    ;;
   *)
    echo "[No text/plain in this mail]"
    ;;
  esac
 fi
}

if [ "$2" = "user" ]; then
 if [ "$3" = "" ]; then
  user="$USER"
else
  user="$3"
fi
 if [ -f /home/$user/.email2text.files ]; then
  while read f; do
   if [ -e $f ]; then
    files="$files $f"
   fi
  done </home/$user/.email2text.files
else
  files=/home/$user/mail/INBOX
fi
else
 files=$2
fi

if [ "$1" = "num" ]; then
 nice -n 5 lsmbox -ot $files | awk '$1=="Summary:" {printf("%d", $3)}'
exit
elif [ "$1" = "info" ]; then
 # wyszukaj nowe wiadomosci i kazda z nich przekaz do skryptu
 # mboxgrep czeka na zakonczenie dzialania uruchomionego skryptu
 # przed przystapieniem do przetwarzania nastepnej wiadomosci
 nice -n 5 mboxgrep --pipe=$0 -v -G -H "^Status:" $files
 exit
elif [ "$1" = "split" ]; then
```

```
 # podziel wiadomosci i przekaz do skryptu
 # formail czeka na zakonczenie dzialania uruchomionego skryptu
 # przed przystapieniem do przetwarzania nastepnej wiadomosci
 formail -c -s $0 < $2
 exit
elif [ "$1" = "header" ]; then
 echo "$2" | base64qp2utf8
exit
elif [ "$1" = "header" ]; then
 echo "USAGE:"
echo "$0 num|info user nazwa_uzytkownika"
echo "  wtedy gdy istnieje czytamy ~/.email2text.files celem uzyskania"
echo "  listy plików do sprawdzenia (jeden plik na linię"
echo "  domyśnie brany /home/nazwa_uzytkownika/mail/INBOX"
echo "$0 num|info|split|header plik"
exit
fi

script_name=`basename $0`
if [ "$script_name" = "mail2text-long" ]; then
 MESSAGE=`head -n 300`
email2text "$MESSAGE"
exit
else #if [ "$script_name" = "mail2text-short" ]; then
 MESSAGE=`head -n 300`
echo -e "************************************************************"
email2text "$MESSAGE" | head -n 8
 echo -e "************************************************************"
exit
fi
```

SMTP2XMPP.PL

```
#!/usr/bin/perl -w

# uwaga dane wejściowe do programu pochodzące z maila (wydobyte np. przy pomocy formail)
# warto przepuścić przez mail2text.sh celem dokonania konwersji 7 lub 8 bitowego kodowania
# znaków narodowych do utf-8

use open ":encoding(utf8)";
use open ":std";

# konfiguracja
$SERVER = "jabber.moja.domena";
$PORT = 5222;
$TLS = 1;
$USER = "mail-system";
$PASS = "secret";

require "/etc/xmmp-mail-system-conf.pl";

my $RES = "New Mail Notify";
my $SUBJECT = "";
my $TYPE = "headline"; # "headline", "chat" or ""

# XEP-0160: Best Practices for Handling Offline Messages:
# "Messages with a 'type' attribute whose value is "headline" SHOULD NOT be stored offline,
#    since such messages are usually time-sensitive."
```

```
# deklaracje wstępne
my $message="";
use Getopt::Long;
GetOptions ("to=s" => \$to_jid, "message=s" => \$message, "type=s"  => \$TYPE, "subject=s" => ⤸
\$SUBJECT);

if ($message eq "") {
 while(<STDIN>) {
  $message .= $_;
 }
}

use Net::XMPP;
my $cli = new Net::XMPP::Client();

# łączenie
my $con = $cli->Connect(hostname=>$SERVER, port=>$PORT, tls=>$TLS);
if (!(defined($con))) {
 print "Nie można połączyć się z serwerem\n";
 exit(-1);
}

# logowanie
my @aut = $cli->AuthSend(username=>$USER, password=>$PASS, resource=>$RES);
if ($aut[0] ne "ok") {
 print "Błąd autoryzacji";
 exit(-2);
}

# wysylanie
$cli->MessageSend(to=>$to_jid, subject=>$SUBJECT, body=>$message, type=>$TYPE, from=>$USER . ⤸
"@" . $SERVER);

# zamkniecie polaczenia i zakonczenie
$cli->Disconnect;
exit(0);
```

XMPP2SMTP.SH

```
#!/bin/bash

#/usr/bin/psql "select xml from spool where username=\"$1\""


if [ -f /var/lib/ejabberd/xmpp2smtp.seq.txt ]; then
 MIN_SEQ=`cat /var/lib/ejabberd/xmpp2smtp.seq.txt`
fi
MIN_SEQ=${MIN_SEQ:-0}

MAX_SEQ=`echo "select MAX(seq) from spool" | /usr/bin/psql -t | tr -d ' '`
MAX_SEQ=${MAX_SEQ:-0}

for user in $1; do
 SQL="select xml from spool where seq > $MIN_SEQ and seq <= $MAX_SEQ and username='$user'"
( echo "<offline_message>"
```

```
    echo "$SQL" | /usr/bin/psql -t
    echo "</offline_message>"
) | xalan -xsl `dirname $0`/xmpp2smtp.xsl 2> /dev/null | mail -E -s "XMPP Notify" $user
done

echo $MAX_SEQ > /var/lib/ejabberd/xmpp2smtp.seq.txt
```

xmpp2smtp.xsl

```
<?xml version="1.0"?>

<!--
Copyright (c) 2010, Robert Paciorek (http://www.opcode.eu.org/), BSD-type license

Redystrybucja wersji źródłowych i wynikowych, po lub bez dokonywania
modyfikacji JEST DOZWOLONA, pod warunkiem zachowania niniejszej informacji o
prawach autorskich. Autor NIE ponosi JAKIEJKOLWIEK odpowiedzialności za skutki
użytkowania tego dokumentu/programu oraz za wykorzystanie zawartych tu informacji.

This program is free software. Redistribution and use in source and binary
forms, with or without modification, ARE PERMITTED provided save this copyright
notice. This document/program is distributed WITHOUT any warranty, use at YOUR
own risk.
-->

<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fn="http://www.w3.org/2005/xpath-functions"
xmlns:d="urn:xmpp:delay"
exclude-result-prefixes="fn"
>
 <xsl:output method='text' encoding='utf-8' />

<xsl:variable name="LF"><xsl:text>
</xsl:text></xsl:variable>

<xsl:strip-space elements="*" />

<xsl:template match="message">
  <text>IM from:    </text>
  <xsl:value-of select="@from"/>
  <xsl:value-of select="$LF"/>
  <xsl:apply-templates select="subject"/>
  <xsl:apply-templates select="body"/>
  <xsl:apply-templates select="d:delay"/>
  <xsl:value-of select="$LF"/>
 </xsl:template>

<xsl:template match="subject">
  <text>    subject: </text>
  <xsl:value-of select="."/>
  <xsl:value-of select="$LF"/>
 </xsl:template>
 <xsl:template match="body">
  <text>    body:    </text>
  <xsl:value-of select="."/>
  <xsl:value-of select="$LF"/>
 </xsl:template>
```

```
  <xsl:template match="d:delay">
   <text>    date:     </text>
   <xsl:value-of select="@stamp"/>
   <xsl:value-of select="$LF"/>
  </xsl:template>
</xsl:stylesheet>
```

MAIL-BOT.PL

```
#!/usr/bin/perl -w

# konfiguracja
$SERVER = "jabber.moja.domena";
$PORT = 5222;
$TLS = 1;
$USER = "mail-system";
$PASS = "secret";

require "/etc/xmmp-mail-system-conf.pl";

$RES = "Mail Info System";

# deklaracje wstepne
use Net::XMPP;
use open ":encoding(utf8)";
use open ":std";
use POSIX qw(strftime);


# obsługa zakonczenia
$SIG{HUP} = \&End;
$SIG{KILL} = \&End;
$SIG{TERM} = \&End;
$SIG{INT} = \&End;


# tworzenie klienta
my $bot = new Net::XMPP::Client();
$bot->SetCallBacks(message=>\&ReciveMessage, presence=>\&RecivePresence);


# łączenie
my $con = $bot->Connect(hostname=>$SERVER, port=>$PORT, tls=>$TLS);
if (!(defined($con))) {
 print "Nie można połączyć się z serwerem\n";
 exit(-1);
}


# logowanie
my @aut = $bot->AuthSend(username=>$USER, password=>$PASS, resource=>$RES);
if ($aut[0] ne "ok") {
 print "Błąd autoryzacji";
 exit(-2);
}

# Roster i Presence
```

```
$bot->RosterGet();
$bot->PresenceSend();

# petla oczekiwania
while(defined($bot->Process())) { }

exit(0);




sub End{
 # zamkniecie polaczenia i zakonczenie
 $bot->Disconnect;
 exit(0);
}

sub ReciveMessage
{
 my $sid = shift;
 my $message = shift;

if ($message->GetFrom("jid")->GetServer() ne $SERVER) {
  $bot->Send($message->Reply(body=>"Tylko dla uzytkownikow z $SERVER"));
  return;
 }

my $from = $message->GetFrom("jid")->GetUserID();
 my $body = $message->GetBody();

if ($body eq "num") {
  # lsmbox -ots mail/INBOX | awk '$1=="mail/INBOX" {print $2}'
  my $wynik="";
  open(DATA, "mail2text num user $from | ");
  while (<DATA>) {$wynik .= $_;}
  close(DATA);

  $bot->Send($message->Reply(body=>"Masz $wynik NOWYCH wiadomosci"));
 } elsif  ($body eq "info") {
  $bot->Send($message->Reply(body=>"prosze czekac ..."));

  my $wynik="Streszczenia otrzymanych wiadomosci:\n";
  open(DATA, "mail2text info user $from | ");
  while (<DATA>) {$wynik .= $_;}
  close(DATA);

  $bot->Send($message->Reply(body=>"$wynik"));
 } elsif  ($body eq "time") {
  my $czas = time;
  my $wynik="CZAS:\n";
  $wynik .= " time stamp: $czas\n";
  $wynik .= " utc:        " . strftime("%Y-%m-%d %H:%M:%S +0000 (UTC)", gmtime($czas)) . "\n";
  $wynik .= " local:      " . strftime("%Y-%m-%d %H:%M:%S %z (%Z), %A", localtime($czas)) . ⤸
  "\n";
  $bot->Send($message->Reply(body=>"$wynik"));
 } else {
  my $wynik="XMPP New Mail Info Bot:\n\n";
  $wynik .= "num = ilosc nowych wiadomosci pocztowych\n";
  $wynik .= "info = streszczenie z wiadomosci\n";
  $wynik .= "time = aktualny czas\n";
```

```
  $wynik .= "help = ten komunikat\n\n";
  $wynik .= "Sprawdzane sa tylko wiadomosci w ~/mail/INBOX\n\n";
  $wynik .= "W celu przyspieszenia pracy nalezy przenosic starsze wiadomosci";
  $wynik .= "z INBOX do innych folderow pocztowych";
  $bot->Send($message->Reply(body=>"$wynik"));
 }
}


sub RecivePresence {
 my $sid = shift;
 my $presence = shift;

my $type = $presence->GetType();

# pozwalamy dodawc i usuwac siebie z listy kontaktow
 if ($type eq "subscribe") {
  $bot->Send($presence->Reply(type=>'subscribed'));
 } elsif ($type eq "unsubscribe") {
  $bot->Send($presence->Reply(type=>'unsubscribed'));
 }

print "00000";
}
```

JID ALIASES (EJABBERD)

Pomimo podobieństw w sposobie adresacji do poczty elektronicznej, w przypadku XMPP trudno jest wymyślić działanie aliasów analogicznych do aliasów pocztowych. Problemem jest czy i jak z aliasów ma być wysyłana widoczność - jeżeli tak samo jak z konta głównego to część zastosowań traci sens oraz to czy ma być wspólna lista kontaktów dla aliasów i konta głównego czy rozdzielne - przy wspólnej także niektóre zastosowania aliasów tracą sens. Natomiast jeżeli zdecydujemy się na oddzielną prezentację i oddzielne listy kontaktów to dochodzimy do przedstawionego tutaj rozwiązania kilku kont o wspólnym haśle i opcjonalnym przekazywaniu wiadomości off-line lub ich kopii na konto główne. W przypadku nielogowania się na takie konta i korzystania z przekazania wiadomości offline, mamy analogiczne rozwiązanie jak opracowane prze zemnie metody uzyskania aliasów w jabberd i jabberd2 (szukaj w archiwum). Warto także zainteresować się stanowiącym część ejabberd'a "mod_shared_roster"

Kontami podstawowymi są konta o nazwach odpowiadających kontu shell'owemu (autoryzacja PAM), ponadto dla każdego z takich kont określona jest domena podstawowa. Aliasy tworzone są na podstawie pliku tekstowego zawierającego wpisy postaci:

`aliasname aliasdomain realname realdomain aliasflag`

Gdzie można używać * po lewej stronie (dowolny użytkownik/domena) i % po prawej stronie (podstawienie dopasowywanej nazwy nazwy użytkownika/domeny), natomiast aliasflag określa na które z kont mają trafiać wiadomości off-line.

Konta aliasowe umożliwiają logowanie, posiadają własny roadster, natomiast presence wysyłany jest tylko w przypadku zalogowania bezpośrednio na konto aliasowe. Dokładniejszy opis w skrypcie przetwarzającym ten plik na potrzeby ejabberda - ejabberd_alias_check.sh (w skrypcie tym należy także ustalić ścieżkę do pliku aliasów). Na system oprócz wspomnainego skryptu składają się dwa patch'e:

- `ejabberd_alias_check.diff` - umożliwiający logowanie się na aliasowe konta definiowane w stosownym pliku, patch jest przystosowany do łatania http://ejabberd.jabber.ru/pam można go jednak dostosować do innych modułów autoryzacji, patch wymaga "ejabberd_alias_check.sh" i podania ścieżki dostępowej do niego w kodzie
- `ejabberd_alias_forward.diff` - implementuje opcjonalne przekazywanie oryginału lub kopii wiadomości otrzymanej na alias na konto podstawowe, docelowo ma być zastąpiony AMP (XEP-0079), patch wymaga "ejabberd_alias_check.sh" i podania ścieżki dostępowej do niego w kodzie

Polecam także mój system powiadomień XMPP/SMTP zawierający m.in. łatką "ejabberd_log_offline_message.diff" powodującą zapisywanie wiadomości trybu offline także do plików tekstowych, które wykorzystuje w moim systemie do przesyłania powiadomień o nowych IM na e-mail.

EJABBERD_ALIAS_CHECK.SH

```
#!/bin/bash

ALIAS_FILE='/etc/ejabberd/aliases'

# FILE format:
# aliasname aliasdomain realname realdomain aliasflag
#
# aliasname == * => any user in this domain
# aliasdomain == * => this user in any domain
# realname OR realdomain == % use orginal aliasname OR aliasdomain
# aliasflag == real (OR not set for exist alias) => move msg to real user
# aliasflag == copy => copy msg to real user
# aliasflag == alias => save message only for alias user

read tmp
read user serwer;

real_user=`awk -v name=$user -v host=$serwer '
 BEGIN {
  ctrl=0
  real_user=name
  real_host=host
  flag=""
 }
 $1==name && $2==host {
  real_user=$3
  real_host=$4
  flag=$5
  ctrl=3
  nextfile # end of parsing file
 }
 $1=="*" && $2==host && ctrl<1 {
  real_user=$3
  real_host=$4
  flag=$5
  ctrl=1
 }
 $1==name && $2=="*" && ctrl<2 {
  real_user=$3
  real_host=$4
  flag=$5
  ctrl=2
 }
 END {
  if (real_user == "%")
   real_user=name
  if (real_host == "%")
   real_host=host
  if (flag == "") {
   if (ctrl == 0)
    flag="alias"
   else
    flag="real"
  }
  printf("%s %s %s", real_user, real_host, flag)
 }' $ALIAS_FILE`

#echo "name=$user host=$serwer" >&2
```

```
#echo "$real_user $ALIAS_FILE" >&2

printf "\x0\x"$(printf "%X" $(( `echo ${real_user} | wc -c` - 1 )) )
# wc a nie ${#ala} bo wc liczy bajty a nie znaki (UTF-8)
echo "${real_user}"
```

VoIP, SIP i Asterisk

Kolejną przydatną usługą są połączenia głosowe realizowane przez protokół IP. Jest to chyba najbardziej wymagająca dla tego protokołu usługa, gdyż wymaga ona bardzo małych opóźnień (dane spóźnione są bezużyteczne, a nie możemy też sobie pozwolić na duży bufor, gdyż usługa realizowana jest w czasie rzeczywistym). Jednak pomimo tych problemów telefonia IP (VoIP[1707]) ma liczne zalety - takie jak elastyczność, wysoka konfigurowalność i niskie koszta połączeń (nie jest potrzebny fizyczny kanał a tylko pasmo transmisji danych). Chyba najpopularniejszą softwarową centralką telefoniczną, obsługującą nie tylko VoIP ale także klasyczną telefonię jest Asterisk[1708]. Oprogramowanie to jest ponadto, w stanie rozwiązać problemy protokołu SIP[1709] z NAT[1710]em (wystarczy żeby serwer asteriska miał IP z za NATu i publiczne).

Prezentuję tutaj zestaw najistotniejszych plików konfiguracji Asteriska obejmujący:

- konfigurację dialplanu czyli systemu numeracyjnego i sposobów zestawiania połączeń:
  - extensions.lua - podstawoiwe definicje dialplanu stworzonego w lua (w tym inkludowanie elementów z `extensions.lua.d`) - ta metoda konfiguracji wydaje się najprzyjemniejsza (dość dobra metodą jest też AEL, natomiast najbardziej standardowe definiowanie dilplanu w extensions.conf jest upiorne ze względu na problematyczność / nieczytelnośc konstrukcji takich jak pętle, warunki, itd)
  - extensions.ael - kontekst `after_dial`, który jest wywoływany poprzez G() z komendy Dial() i ze względu na specyfikę tego wywołania (skok do priorytetu 2) nie może być zdefiniowany w lua która nie wspiera priorytetów w dialplanie
  - extensions.lua.d/call.lua - podstawowe funkcje związane z realizacją połączeń (dzwonienie na zwenątrz, dzonienie do zarejestrowanych użytkowników z wsparciem dla vhostów, nagrywanie rozmów, wyszukiwanie ENUM[1711])
  - extensions.lua.d/conference.lua - funkcje związane z tworzeniem i przyłączaniem się do konferencji
  - extensions.lua.d/ivr.lua - funkcje pomocnicze związane z realizacją systemów IVR
  - extensions.lua.d/transfers.lua - kontekst `input_from_transfers` do którego trafiają z `after_dial` realizowane transfery (zajmuje się on identyfikacją wykonującego transfer i przeniesieniem do jego kontekstu wychodzącego) oraz funkcje pomocnicze związane z realizacją transferów
  - extensions.lua.d/addresses.lua - specyfikacja adresów dla połączeń przychodzących (m.in. przykładowe IVR, definicja dzwonienia na numery konferencyjne, numer do parkowania rozmów, powiadomienia jabberowe o przychodzącej rozmowie)
  - extensions.lua.d/clients.lua - specyfikacja realizacji połączeń od zautoryzwoanych klientów (peerów)
- konfigurację akcji podejmowanych w trakcie rozmowy - features.conf
- konfigurację kont sip - sip.conf (plik ten dodatkowo inkluduje `sip-register.conf`, `sip-phone.conf` oraz `sip-provider.conf` zawierające lokalną konfigurację kont)
- konfigurację poczty głosowej - voicemail.conf (plik ten dodatkowo inkluduje `voicemail-local.conf` zawierający lokalną konfigurację kont)
- konfigurację XMPP[1712] oraz jingle[1713] (gtalk):
  - jabber.conf - ogólna konfiguracja xmpp, adres do wysyłania powiadomień, adresy do odbierajnia rozmów jingle (plik ten dodatkowo inkluduje `jabber-users.conf` zawierające lokalną konfigurację kont)
  - jingle.conf - ogólna konfiguracja jingle, konfiguracja połączeń wychodzących jingle (plik ten dodatkowo inkluduje `jingle-outcoming.conf` zawierające lokalną konfigurację kont)
  - gtalk.conf powinien być dowiązaniem do tego pliku

Warto włączyć rejestrowanie w logu CDR (`/var/log/asterisk/cdr-csv/Master.csv`) nieudanych prob dzwonienia poprzez opcję `unanswered = yes` w `/etc/asterisk/cdr.conf` oraz rejestrowanie przebiegu dzwonienia w `/var/log/asterisk/verbose` poprzez następujące wpisy w `/etc/asterisk/logger.conf`:

```
[logfiles]
        verbose => verbose
        debug => debug
messages => notice,warning,error
```

Sporym wyzwaniem bywa nie rzadko także odpowiednia konfiguracja kodeków. Podstawowym kodekiem jest G.711[1714] (w dwóch odmianach ulaw / alaw) - zapewnia dobrą jakość, ale wymaga dużego pasma, oprócz niego godnymi zainteresowania jest kilka wolnych kodeków takich jak: G.726[1715], Speex[1716], gsm[1717], iLBC[1718], G.722[1719]. Ponadto standardowy Asterisk w chwili obecnej nie wspiera integracji z kontami shellowymi użytkowników, ale można to uzyskać z wykorzystaniem bazy LDAP.

Zobacz w Sieci: voip-info[1720].

EXTENSIONS.LUA

```
——————————————
–                                          –
–             USTAWIENIA OGÓLNE            –
–                                          –
——————————————

– każemy szukać włączanych plików (funkcje fragmenty dialplanu itd)
– w /etc/asterisk/extensions.lua.d/ w postaci plików .lua
package.path = package.path .. ";/etc/asterisk/extensions.lua.d/?.lua"

– definiujemy (na razie) pustą tablicę z dialplanem
extensions = {}
– oraz pomocnicze zmienne dla dialplanu ...
any_ext = "_[0-9a-zA-Z.*#+-]."

–   poziom na którym wypisywane są komunikaty
verb=2
–   lista sprawdzanych sieci ENUM
enum_networks = { "e164.arpa", "e164.org", "e164.info", "freenum.org" }


——————————————
–                                          –
–      BIBLIOTEKI FUNCJI DIALPLANU         –
–                                          –
——————————————

– standardowa aplikacja "demo" - kontekst "demo_lua" i odpowiednie funkcje
require("demo")

– funcje "call_local_address", "call_address" oraz "call_record"
require("call")

– funkcje "conference_create" i "conference_enter"
require("conference")

– funcje "ivr_play_and_read" oraz "ivr_play_and_goto"
require("ivr")

– obsługa transferów
require("transfers")

——————————————
–                                          –
–              NASZE ADRESY                –
–                                          –
——————————————

– spis obsługiwanych domen
– wpisy dodawane jako elementy tablicy postaci:
```

```
--          local_domains["DOMENA"] =1
local_domains = {}

-- połączenia przychodzące na poszczególne adresy
-- każdy adres definiowany w postaci kontekstu:
--      * dla połączeń wewnętrznych (od zautoryzwoanych klientów)
--              extensions["local_input_for/USER@DOMENA"] = { ... }
--      * dla połączeń zewnętrznych i wewnętrznych (jeżeli nie ma
local_input_for/USER@DOMENA)
--              extensions["input_for/USER@DOMENA"] = { ... }

-- definiujemy kontekst "addresses" do którego będą trafiały rozmowy
-- przychodzące na adresy nie posiadające własnych kontekstów
extensions.addresses = {}
-- UWAGA: w ramach każdego z extentions w tym kontekscie:
--    1. trzeba zadbać o włączenie nagrywania poprzez wywołanie:
--              call_record(exten, "incoming")
--    2. można sprawdzać skąd przychodzi połączenie poprzez:
--              channel["CALL_TYPE"]:get()
--          zmienna CALL_TYPE jest ustawiona na:
--          * local    - dla połączeń od zautoryzowanych klientów
--          * public - dla połączeńprzychodzących z zewnątrz

require("addresses")

-- ustawiamy obsługę błędnych adresów w kontekscie addresses
extensions.addresses.i = function(context, exten)
 app.Verbose(verb, "extention \"".. channel["INVALID_EXTEN"]:get()
  .."\" not found in context \"addresses\"")
 app.Hangup(1)
end


----------------------------------------------------------------
--                                                            --
--                       NASI KLIENCI                         --
--                                                            --
----------------------------------------------------------------


-- połączenia od klientów zewnętrznych (niezautoryzowanych)
-- mają prawo:
--    * kontaktu z naszymi użytkownikami
--    * korzystania z usług publicznych
extensions.incoming_from_external = {
 [any_ext] = call_local_address;
};

-- połaczenia od klientów zautoryzowanych
-- identyfikujemy typ adresu (lokalny, zdalny sip, zdalny ENUM, zdalny PSTN, ..)
-- wykonujemy połączenie na stosowny numer
extensions.incoming_from_auth_user = {
 [any_ext] = call_addres;
};

-- konteksty połączeń przychodzących poszczególnych klientów sip
-- (do nich prowadzą wpisy w sip.conf) postaci:
--        extensions["input/LOGIN"] = { include = {"incoming_from_auth_user"}; }
-- typowo inkludujące "incoming_from_auth_user",
-- ale mogące także definiować prefixy itp
```

– konteksty połączeń wychodzących poszczególnych klientów sip postaci:
– extensions["output/LOGIN"] = { [any_ext] = call_local_address; }
– typowo oprócz obsługi lokalnych adresów (patrz sekcja "nasze adresy")
– realizowanej poprzez funkcję call_local_address zawierają definicje
– wyjść przez operatorów SIP i indywidualną listę adresową
– gdy korzystamy z funkcji "call_addres" to
– każdy kontekst "input/LOGIN" musi mieć odpowiedni "output/LOGIN"

```
require("clients")
```

ADDRESSES.LUA

```
default_domain = "opcode.eu.org"
local_domains["opcode.eu.org"] = 1


-- na potrzeby jingle / gtalk
extensions["input_for/rrp@voip.opcode.eu.org@opcode.eu.org"] = {
 s = function()
  app.goto("input_for/rrp@opcode.eu.org", "s", "1")
 end;
}

extensions["input_for/rrp@opcode.eu.org"] = {
 s = function(context, exten)
  -- w tym miejscu mogą być umieszczone reguły odpowiedzialne za
  -- filtrowanie połączaeń oraz wybór wywoływanych linii
  channel["REAL_DIAL1_TO"] = "SIP/rrp1&SIP/rrp2"
  channel["REAL_DIAL2_TO"] = ""
  return app.GoTo("1", 1)
 end;
 ["1"] = function()
  app.Verbose(verb, "Do Dial() to " .. channel["REAL_DIAL1_TO"]:get())
  app.JabberSend("notify",  "rrp@opcode.eu.org",
        "Połączenie VoIP od: " .. channel["CALLERID(all)"]:get() )
  app.Dial(channel["REAL_DIAL1_TO"]:get(), 40, "G(after_dial^s^1)")

  -- jako że Dial() z G() nie wraca w przypadku uzyskania połączenia
  -- to dojście do tego miejsca oznacza że dzwonienie nie udało się
  -- realizujemy dzwonienie w drugim wariancie (jeżeli jest ustawione)
  if channel["REAL_DIAL2_TO"]:get() ~= "" then
   app.Verbose(verb, "Do (second) Dial() to "
    .. channel["REAL_DIAL1_TO"]:get())
   app.Dial(channel["REAL_DIAL2_TO"]:get(), 40, "G(after_dial^s^1)")
  end

  -- gdy też się nie udało (lub go nie było) to:
  --   1) ustalamy powód niepowodzenia Dial()
  --      inormacje o nim zawierają zmienne: DIALSTATUS i HANGUPCAUSE
  status1 = channel["DIALSTATUS"]:get()
  status2 = channel["HANGUPCAUSE"]:get()
  app.Verbose(verb, "Dial end with DIALSTATUS: " .. status1
   .. " and HANGUPCAUSE: " .. status2)

  if status1 == "CHANUNAVAIL" then
```

```
  – Żaden telefon nie jest zalogowany.
  dial_error="unavail"
 elseif status1 == "NOANSWER" then
  – Żadna linia nie została odebrana
  dial_error="no_answer"
 elseif status1 == "BUSY" and status2 == "21" then
  – Połączenie zostało odrzucone.
  dial_error="reject"
 elseif status1 == "BUSY" and status2 == "17" then
  – Wszystkie linie są zajęte.
  dial_error="busy"
 else
  – Połączenie nie może zostać zrealizowane.
  dial_error="error"
 end

 channel["Dial_Error"] = dial_error

 –   2) przechodzimy do systemu IVR
 return app.GoTo("ivr_after_dial", 1)
end;
["ivr_after_dial"] =function()
 app.Answer()
 ivr_play_and_goto(
  – Tu numer 222, 443, 895.
  "rrp/hello"
  – {wiadomość określona w oparciu o zmienną Dial_Error}
  .. "&rrp/dial_".. channel["Dial_Error"]:get()
  – Proszę wybrać 1 aby ponowić próbę dzwonienia,
  – lub wybrać 2 aby zostawić wiadomość, lub rozłączyć się.
  .. "&rrp/ivr_intro",
  6,
  – Brak wyboru - rozłączam.
  "rrp/ivr_hangup"
 )
end;
["2"] = function()
 ivr_play_and_goto(
  – Jeżeli akceptujesz zasady kontaktu telefonicznego przedstawione
  – pod adresem www.opcode.eu.org/bercik i twoja wiadomość spełnia
  – określone tam kryteria proszę wybrać 3.
  – Prezentacja zasad dostępna także po wybraniu 9.
  "rrp/ivr_intro2",
  6,
  – Brak akceptacji zasad - rozłączam.
  "rrp/ivr_no_recording"
 )
end;
["3"] = function()
 – Zasady zostały zaakceptowane. Proszę zostawić wiadomość po sygnale.
 app.Playback("rrp/ivr_recording")
 app.Voicemail("1001", "s")
 app.Hangup()
end;
["9"] = function()
 – Preferowanym sposobem kontaktu jest poczta elektroniczna -
 – robert@opcode.eu.org. Odbycie rozmowy telefonicznej na żywo możliwe
 – jest po wcześniejszym umówieniu się przy pomocy poczty elektronicznej
 – (lub jeżeli dopisze szczęście i akurat odbiorę).
 – W pozostałych przypadkach (jeżeli sprawa jest ważna / niecierpiąca
```

```
  -- zwłoki a nie masz możliwości wysłania e-mail'a) możliwe jest
  -- zostawienie wiadomości głosowej:
  --     * Przedstaw się.
  --     * Opisz dokładnie sprawę w której dzwonisz (nie zostawiaj informacji
  --         typu "Proszę o kontakt" bez dodatkowych wyjaśnień / opisu sytuacji
  --         i tak dalej).
  --     * Jeżeli oczekujesz kontaktu zwrotnego pozostaw swój adres poczty
  --         elektronicznej (jest to jedyna forma kontaktu zwrotnego).
  -- Wiadomości głosowe nie spełniające powyższych kryteriów oraz
  -- wiadomości głosowe w innych językach niż polski będą ignorowane.
  app.Playback("rrp/ivr_rules")

  ivr_play_and_goto(
   -- Jeżeli akceptujesz powyższe zasady i chcesz zostawić wiadomość
   -- spełniającą określone tam kryteria proszę wybrać 3.
   "rrp/ivr_intro3",
   6,
   -- Brak akceptacji zasad - rozłączam.
   "rrp/ivr_no_recording"
  )
 end;
 i = function()
  app.Playback("pbx-invalid")
  return app.GoTo("ivr_after_dial", 1)
 end;
};

extensions["input_for/demoX@opcode.eu.org"] = {
 include = {"demo_lua"}
};

extensions["input_from/sipprovider2"] = {
 s = function()
  channel["CALLERID(all)"] = channel["CALLERID(name)"]:get()
   .. " @ sipprovider2 <00" .. channel["CALLERID(num)"]:get() .. ">"
  return app.goto("input_for/rrp@opcode.eu.org", "s", "1")
 end;
}


-- konferencje:
--   tworzenie tylko dla zalogowanych klientów
--   dołączanie dla każdego kto zna numer i pin
extensions["local_input_for/7900@opcode.eu.org"] = {
 s = conference_create;
};

extensions["addresses"] = {
 ["_77!"] = function()
  app.Park()
 end;
 ["_78XX!"] = function(context, exten)
  app.ParkedCall(string.sub(exten,1,4))
 end;

["_79XX@opcode.eu.org"] = function(context, exten)
  -- przyłączenie do konferencji możliwe jest poprzez numer
  -- tworzony na zasadzie prefix . numer_konferencji
  -- w tym wypadku prefixem jest 79,
  -- a numer konferencji jest zawsze dwu cyfrowy
```

```
  conference_enter(string.sub(exten,3,4))
 end;

["_74!"] = function()
  app.goto("input_for/demoX@opcode.eu.org", "s", "1")
 end;
}
```

CALL.LUA

```
– pomocnicza funkcja sprawdzajaca czy istnieje podany context
– i zapisujaca info do logu
function context_exists(name)
 if channel["DIALPLAN_EXISTS("..name..")"]:get() == "1" then
  app.Verbose(verb, "Context " ..name.. " exists")
  return true
 else
  app.Verbose(verb, "Context " ..name.. " don't exist")
  return false
 end
end

– pomocnicza funkcja realizująca dzwonienie enum
function call_enum(exten)
 for i,net in ipairs(enum_networks) do
  app.Verbose(verb, "Trying ".. exten .." in enum network: ".. net)
  cnt = channel["ENUMLOOKUP(+".. exten ..",ALL,c,,".. net ..")"]:get()
  for j=cnt, 0, -1 do
   uri = channel["ENUMLOOKUP(+".. exten ..",ALL,,".. j ..","
    .. net ..")"]:get()
   pos = string.find(uri, ":")
   if pos ~= nil then
    proto = string.sub(uri, 0, pos-1)
    addr = string.sub(uri, pos+1)
    app.Verbose(verb, "Find proto=".. proto .." addr=".. addr)
    if proto == "sip" then
     app.Dial("SIP/" .. addr, nil, "G(after_dial^s^1)")
    elseif proto == "iax2" then
     app.Dial("IAX2/" .. addr, nil, "G(after_dial^s^1)")
    elseif proto == "h323" then
     app.Dial("H323/" .. addr, nil, "G(after_dial^s^1)")
    end
    if channel["DIALSTATUS"]:get() == "ANSWER" then
     return true
    end
   end
  end
 end
 return false
end

– funkcja aktywująca nagrywanie (generuje nazwę pliku i wywołuje Monitor())
function call_record(user, way)
 if channel["is_recording_for_" .. user]:get() ~= "1" then
  channel["is_recording_for_" .. user] = "1"

  filename = "/home/voip/" .. way .. "/" .. user .. "/"
```

```
    .. os.date("%Y-%m-%d_%H:%M:%S_")
    .. string.gsub(channel["CHANNEL"]:get(), "/", "_")

  app.Verbose(verb, "Start recording for " .. user
    .. " as " .. way .. " dial, to filename " .. filename)

  –app.MixMonitor(filename .. ".wav");
  app.Monitor("gsm",filename);
 end
end



– funkcja realizująca dzwonienie pod lokalne adresy
– z rozróżnieniem publiczności lub nie tych adresów
function call_local_address(context, exten, call_type, domain, channel_type)
 – parametry nieobowiazkowe i ich wartosci domyslne
 if channel_type == nil then
  channel_type = channel["CHANNEL(channeltype)"]:get()
 end
 if domain == nil and channel_type == "SIP" then
  domain       = channel["SIPDOMAIN"]:get()
 end
 if domain == nil then
  domain       = default_domain
end
 if call_type == nil then
  – zakładamy że konteksty użytkownika mają postać typ/username[@domain]
  –     typ == input_for    => rozmowy przychodzące do usera
  –     typ == input        => rozmowy przychodzące od usera
  –     typ == output       => rozmowy wychodzące od usera
  local pos = string.find(context, "/")
  if pos ~= nil and string.sub(context, 0, pos-1) == "output" then
   call_type = "local"
  else
   call_type = "public"
  end
 end

app.Verbose(verb, "incoming_call type=" .. call_type ..
  " to=" .. exten .. "@" .. domain .. " via=" .. channel_type)

– aby działało nagrywanie (a w niektórych przypadkach samo połączenie)
 – niestety musimy odebrać kanał który wykonuje Dail()
 –
 – najprościejbyłoby to robić przed wykonaniem Dial(), ale:
 – 1. przy rozmowach wychodzących od naszych klientów to nie problem
 –        (i wtedy tak robimy)
 – 2. przy przychodzących to mało szczęśliwe rozwiązanie
 –        (powoduje niespójną sygnalizację u dzwoniącego - połączenie
 –        jest odebrane a sygnał dzwonienia jest generowany sztucznie
 –        oraz nalizczanie opłat za czas oczekiwania na połączenie)
 –
 –        w tym celu wywołujemy:
 – app.Dial("SIP/NASZ_PEER", nil, "G(after_dial^s^1)")
 –
 –        macro after_diall zdefiniowane jest w: extensions.ael
 –        (gdyż Lua nie wspiera priorytetów)
 –        odpowiada ono za odebranie kanałów i ich bridgowanie
 –
```

```
 –      wymaga to uprzedniego ustawienia zmiennej CALLING_CHANNEL
 –      na potrzeby after_dial jako zmiennej dziedziczonej:
 channel["_CALLING_CHANNEL"] = channel["CHANNEL"]:get()

– na potrzeby tranferów potrzebujemy także zachować informację o adresacie
 channel["_CALL_TO_USER"] = exten .. "@" .. domain


if call_type == "local" then
  – sprawdzamy istnienie niepublicznego adresu pasującego do tego wywołania
  – a jeżeli istnieje to aktywujemy nagrywanie i przechodzimy do niego
  local address = "local_input_for/" .. exten .. "@" .. domain
  if context_exists(address) then
   call_record(exten .. "@" .. domain, "incoming")
   return app.goto(address, "s", "1")
  end
 end

– jeżeli istnieje żądany adres (w odpowiedniej domenie) to
 – aktywujemy nagrywanie i przechodzimy do niego
 local address = "input_for/" .. exten .. "@" .. domain
if context_exists(address) then
  call_record(exten .. "@" .. domain, "incoming")
  return app.goto(address, "s", "1")
 end

– wygodnie jest mieć też możliwość definiowania adresów jako
 – kolejnych extentions w ramach jednego kontekstu ...
 channel["CALL_TYPE"] = call_type
return app.goto("addresses", exten .. "@" .. domain, "1")

– jeżeli tam także nie będzie takiego adresu to zostanie wywołany z exten ”i”
 – hangup(1) zwracający informację ”nie ma takiego numeru”
end

– funkcja realizująca dzwonienie pod podany adres (dla zalogowanych userów)
– bezpośrednie w przypadku pełnego adresu SIP lub rekordu ENUM
– w pozostałych przypadkach poprzez kontekst output/username
function call_addres(context, exten, check_enum, from, domain, channel_type)
 – parametry nieobowiazkowe i ich wartosci domyslne
 if channel_type == nil then
  channel_type = channel["CHANNEL(channeltype)"]:get()
 end
 if domain == nil and channel_type == "SIP" then
  domain       = channel["SIPDOMAIN"]:get()
 end
 if domain == nil then
  domain       = default_domain
end
 if from == nil then
  – zakładamy że funkcja wywoływana z kontekstu użytkownika
  – mającego postać input/username
  from         = string.sub(context, string.find(context, "/")+1)
 end
 if check_enum == nil then
  – domyślnie sprawdzamy enum dla numerów złożonych wyłącznie z cyfr
  – i dłuższych niż 4 cyfry
  if string.find(exten, "^[0-9]*$") ~= nil and string.len(exten) > 4 then
   check_enum = true
  else
```

```
    check_enum = false
  end
 end

app.Verbose(verb, "call_addres from=" .. from ..
  " to=" .. exten .. "@" .. domain .. " via=" .. channel_type)

– aktywacja nagrywania rozmowy wychodzącej od usera
 call_record(from, "outcoming")

– zapamiętanie kanału wywołującego Dial() ... patrz uwagi w call_local_address()
 channel["_CALLING_CHANNEL"] = channel["CHANNEL"]:get()

– na potrzeby tranferów potrzebujemy także zachować informację
 – o tym kto wywołuje Dial()
 channel["_CALL_FROM_USER"] = from


– zdalny adres SIP lub gtalk/jingle
 if local_domains[domain] == nil then
  app.Verbose(verb, "remote SIP / JINGLE address")
  return app.Dial("SIP/" .. exten .. "@" .. domain .. "&" ..
                "Gtalk/" .. from .. "/" .. exten .. "@" .. domain,
   nil, "G(after_dial^s^1)")
 end

– ENUM
 if check_enum then
  call_enum(exten)
 end

– lokalny exten, zdefiniowany w kontekście połączeń wychodzących
 – user'a podanego w from
 app.Verbose(verb, "goto exten: " .. exten .. " in: " .. "output/" .. from)
 return app.Goto("output/" .. from, exten, "1")
end
```

CLIENTS.LUA

```
– PEERNAME
extensions["input/PEERNAME"] = {
 include = {"incoming_from_auth_user"};
};
extensions["output/PEERNAME"] = {
 [any_ext] = call_local_address;

["_0X."] = function(context, exten)
  exten2 = string.sub(exten, 2)

  app.Verbose(2, "Dial via sipprovider2 to " .. exten2 )
  app.Dial("SIP/" .. exten2 .. "@sipprovider2")

  if channel["DIALSTATUS"]:get() == "BUSY" then
   return app.Busy()
  else
   return app.Hangup()
  end
```

```
 end;

["demo"] = function()
  app.Goto("demo_lua", "s", "1")
 end;
};
```

CONFERENCE.LUA

```
– dynamiczne tworzenie konferencji wieloosobowej
– konferencja zostanie zakończona po wyjściu ostatniego uzytkownika
– dostęp do menu konfiguracyjnego poprzez #
function conference_create(context, exten)
 for confno=10,99,1 do
  state = channel["DEVICE_STATE(confbridge:".. confno ..")"]:get()
  if state ~= "INUSE" then
   app.Verbose(verb, "Using conference number: ".. confno
    .." (in state: ".. state ..")")
   app.Answer()
   – odczytanie numeru konferencji
   app.Playback("conf-enteringno")
   app.SayDigits(confno)
   – zapytanie o PIN
   app.Read("CONFPIN","conf-getpin",nil,nil,3,10)
   – zapamietanie PINu
   pin = channel["CONFPIN"]:get()
   app.Verbose(verb, "Setting PIN=".. pin)
   channel["DB(conf/".. confno .."/pin)"] = pin
   – wejscie do konferencji
   app.Playback("conf-placeintoconf")
   app.ConfBridge(confno,"sa")
   break
  end
 end
end;

– przyłączenie się do podanbej konferencji
– funkcja przeznaczona do wywoływania w numerach budowanych na zasadzie
–   prefix . numer_konferencji
function conference_enter(confno)
 state = channel["DEVICE_STATE(confbridge:".. confno ..")"]:get()
 app.Verbose(verb, "Using conference number: ".. confno
  .." (in state: ".. state ..")")
 if state == "INUSE" then
  pin_db   = channel["DB(conf/".. confno .."/pin)"]:get()
  app.Answer()
  if pin_db == 0 then
   app.Playback("conf-placeintoconf")
   app.ConfBridge(confno,"s")
  else
   – zapytanie o PIN
   app.Read("CONFPIN","conf-getpin",nil,nil,3,10)
   – weryfikacja PINu
   pin_read = channel["CONFPIN"]:get()
   app.Verbose(verb, "Using PIN=".. pin_read
    .." (conference pin is: "..pin_db ..")")
   if pin_db == pin_read then
    app.Playback("conf-placeintoconf")
```

```
      app.ConfBridge(confno,"s")
    else
      app.Playback("conf-invalidpin")
      app.Hangup(21)
    end
  end
 else
  app.Hangup(1)
 end
end;
```

IVR.LUA

```
function ivr_play_and_read(filename, timeout)
 app.Read("ivr_exten", filename, 1, nil, 1, timeout)
 ivr_exten = channel["ivr_exten"]:get()
 app.Verbose(verb, "DTMF input: " .. ivr_exten)
 if ivr_exten == "" then
  return false
 else
  return ivr_exten
end
end;

function ivr_play_and_goto(filename1, timeout, filename2, context)
 ivr_exten = ivr_play_and_read(filename1, 6)
 if ivr_exten then
  return app.goto(context,ivr_exten,1)
 else
  app.Playback(filename2)
  app.Hangup()
 end
end;
```

TRANSFERS.LUA

```
– pomocnicza funkcja wydobywająca rekurencyjnie DIALEDPEERNUMBER
– z kanłu który realizuje ”attended transfer”
function get_dialedpeernumber(transfer_channel)
 local tmp_channel
local transfer_real_from

– pobieramy z DIALEDPEERNUMBER w wskazanym kanale
 transfer_real_from = channel["IMPORT("
  .. transfer_channel ..",DIALEDPEERNUMBER)"]:get()

if transfer_real_from ~= nil and transfer_real_from ~= "" then
  – jeżeli się udało to kończymy
  return transfer_real_from
else
  – w przeciwnym razie sprawdzamy czy w kanale tym:
  –      jest BRIDGEPEER ?
  tmp_channel = channel["IMPORT("
   .. transfer_channel ..",BRIDGEPEER)"]:get()
```

```
  if tmp_channel ~= nil and tmp_channel ~= "" then
   -- gdy jest to wykonujemy się rekurencyjnie na znalezionym kanale
   return get_dialedpeernumber(tmp_channel)
  end

  --          jest TRANSFERNAME ?
  local tmp_channel = channel["IMPORT("
   .. transfer_channel ..",TRANSFERNAME)"]:get()
  if tmp_channel ~= nil and tmp_channel ~= "" then
   -- gdy jest to wykonujemy się rekurencyjnie na znalezionym kanale
   return get_dialedpeernumber(tmp_channel)
  end
 end
end

extensions.input_from_transfers = {
 [any_ext] = function(context, exten)
  local current_channel     = channel["CHANNEL"]:get()
  local dial_from_channel   = channel["CALLING_CHANNEL"]:get()
  local transfer_channel    = channel["TRANSFERERNAME"]:get()
  local from                = channel["CALL_FROM_USER"]:get()
  local to                  = channel["CALL_TO_USER"]:get()
  local transfer_real_from = ""

  if from ~= nil and dial_from_channel ~= current_channel then
   -- dzwonicy      robi transfer <=> dial_from_channel!=current_channel
   -- dzwonicy      jest klientem <=> from!=nil
   app.Verbose(verb, "Transfer from=" .. from)
   transfer_real_from = from
  elseif to ~= nil and dial_from_channel == current_channel then
   -- odbierajcy robi transfer <=> dial_from_channel==current_channel
   -- odbierajcy jest klientem <=> to!=nil
   app.Verbose(verb, "Transfer to=" .. to)
   -- niestety w zmiennej to mamy adres
   -- a nie nazwę klienta (peer'a) robiącego transfer
   -- zatem musimy ją wyciągnąc z zmiennych kanału
   -- najlepsza do tego celu wydaje się DIALEDPEERNUMBER
   -- która zdaje się że powinna zawierać nazwę peer'a który odebrał
   -- Dial() wykonywany w wyniku połączenia z adresem
   transfer_real_from = channel["DIALEDPEERNUMBER"]:get()
  elseif transfer_channel ~= nil then
   -- transfer wywołany z kanału ${TRANSFERERNAME}
   -- jeżeli wywołujący jest naszym klientem
   -- będzie miał ustawione ${CHANNEL(peername)}
   -- i peer ten będzie miał powiązany ze sobą kontekst
   app.Verbose(verb, "Attended transfer init from channel: "
    .. transfer_channel)

   -- z kanału tego (lub dolinkowanych do niego kanałów) wydobywamy
   -- wartość zmiennej DIALEDPEERNUMBER
   transfer_real_from = get_dialedpeernumber(transfer_channel)
  end

  -- jeżeli mamy nazwę peer'a (zautoryzwoanego klienta)
  -- który wywołał transfer
  if transfer_real_from ~= nil and transfer_real_from ~= "" then
   app.Verbose(verb, "transfer_real_from=" .. transfer_real_from)

   -- znajdujemy jego kontekst wychodzący
   local transfer_context =
```

```
    channel["SIPPEER("..transfer_real_from..",context)"]:get()

   if transfer_context ~= nil and transfer_context ~= "" then
    app.Verbose(verb, "transfer_context=" .. transfer_context)

    -- i idziemy do stosownego exten w tym kontekscie
    app.GoTo(transfer_context, exten, "1")
   else
    app.Verbose(verb, "Can't find transfer_context :-(")
    app.Hangup()
   end
  else
   app.Verbose(verb, "Can't find transfer_real_from :-(")
   app.Hangup()
  end
 end;
};
```

EXTENSIONS.AEL

```
context after_dial {
 s => {
  goto calling|1;
  goto called|1;
 };

calling => {
  Verbose(2, "after_dial for dial calling");
  Answer();
  // musi tutaj wystąpić coś co puści jakąś sygnalizację (audio?) w kanał
  // w oczekiwaniu na zbridge'owanie ... Ringing() wydaje się najlepsze
  Ringing();
  // musi tu być coś co spowoduje czekanie ...
  WaitExten(0.6);
  Hangup();
 };

called => {
  Verbose(2, "after_dial for channel called from: ${CALLING_CHANNEL}");
  Ringing();
  Wait(0.3); // nie wiem dlaczego, ale musimy tu poczekać
  Answer();
  Bridge(${CALLING_CHANNEL}, "tTkK");
  Wait(1);
 };

h => {
  System(sleep 3);
  NoOp();
 };

// dla wszystkich pozostałych numerów (mogą przyjść z funkcji transferu)
 // wracamy do dialplanu w lua
_. => {
  goto input_from_transfers|${EXTEN}|1;
 };
};
```

FEATURES.CONF

```
[general]
; zakres numerow uzywanych do odbierania zaparkowanych rozmow
parkpos => 7800-7899

; czas na odebranie zaparkowanej rozmowy
parkingtime => 300

; Maksymalny odstep (ms) miedzy cyframi prefiksu przejscia do "feature"
featuredigittimeout = 1500

; Maksymalny odstep (sek) miedzy cyframi numeru na ktory transferujemy
transferdigittimeout => 4

[featuremap]
; przekazywanie na inny numer
blindxfer => *1

; przekazywanie na inny numer z wczesniejszym polaczeniem sie z tym numerem
atxfer => *2

; rozlaczenie z numerem na ktory przekazujemy i powrot do przekazywanego (przy atxfer)
; FIXME nie działa powrót :-(
disconnect => *0

; parkowanie rozmowy
parkcall => *3

[applicationmap]
```

VOICEMAIL.CONF

```
; konfiguracja poczty głosowej
; http://www.voip-info.org/wiki/index.php?page=Asterisk+config+voicemail.conf

[general]
; formaty zapisu wiadomosci
format=wav49 ;wav|gsm|wav49

; wysyłaj powiadomienia o poczcie
sendvoicemail=yes

; temat wiadomości
emailsubject=[PBX]: Nowa wiadomosc glosowa ${VM_MAILBOX}

; treść wiadomości
emailbody=Hello\n\nYou have new voicemail message from ${VM_CALLERID}.\nRecord time: ${
VM_DATE}, record long: ${VM_DUR}.\n

; adres poczty do pola FROM: dla powiadomien
serveremail=asterisk
```

```
; format czasu użyty w wiadomości
emaildateformat=%Y-%m-%d %H:%M:%S

; kodowanie wiadomości
charset=UTF-8

; załączaj do maila plik dźwiękowy z zapisem zostawionej wiadomości
attach=yes

; usuwaj wiadomości po wysłaniu powiadomienia
; uwaga: opcja ta zdeklarowna w tym miejscu może nie działać
; należy podawać ją w opcjach każdego z użytkowników jako delete=1
delete=yes

; maksymalna ilość zalogowanych równocześnie urzytkowników
maxlogins=3

; maksymalna ilość wiadomości w skrzynce
maxmsg=20

; maksymalna długość wiadomości (w sek.)
maxmessage=180

; minimalna długość wiadomości powodująca jej pozostawienie (w sek.)
;minmessage=3

; maksymalna długość wiadomości powitalnej (w sek.)
;maxgreet=60

; długość ciszy powodująca przerwanie nagrywania (0 - wyłącza)
;maxsilence=0

; spis skrzynek - kontekst domyślny
[default]
; inkludujemy spis skrzynek, wpisy są postaci:
;   nazwa/numer => haslo,opis,email_dla_powiadomien,pager_email,dodatkowe_opcje_rozdzielane|
; typowo:
;   USERID => SECRET,,ADRES@EMAIL,,delete=1
#include <voicemail-local.conf>
```

SIP.CONF

```
; konfiguracja kont SIP
; http://www.voip-info.org/wiki/index.php?page=Asterisk+config+sip.conf

[general]
; słuchanie na porcie 5060 po ipv4 i ipv6 (wspólny socket) po udp, tcp, tsl
bindport=5060
udpbindaddr=::
tcpenable=yes
tcpbindaddr=::
tlsenable=yes
tlsbindaddr=::

; połączenia przychodzące po SIPie trafią do kontekstu "incoming_from_external"
; dla połączeń od naszych klientów ustawienie to zostanie nadpisane
```

```
; zatem do tego kontekstu trafią rozmowy przychodzące od "obcych"
context=incoming_from_external

; używamnie rekordów SRV dla połączeń wychodzących
srvlookup=yes

; wspieramy połaczenia wideo
videosupport=yes

; akceptujemy wszystkie kodeki
allow=all

; DTMF zgodnie z RFC2833
dtmfmode=rfc2833

; nie pozwalamy na przejście na połączenia bezpośrednie p2p
; między klientami (z pominięciem PBX)
;
; nawet gdy canreinvite == yes to wystąpienie jakiejkolwiek z opcji TKWtkw w Dial()
; blokuje możliwość przejścia na połączenioe p2p
canreinvite=no
directmedia=no

; sprawdzamy obecność klienta co 3s
qualify=3000

; dość szybkie wygasanie rejestracji - w trakcie tego czasu (wartość w sekundach)
; dla rozmów idących do peer'a jest on identyfikowany w oparciu o ip, czyli
; aby mógł odebrać rozmowę nie musi się autoryzować
maxexpirey=60
defaultexpirey=60

; nie potrzebujemy wsparcia dla NAT
nat=no


; domena dodawana do identyfikacja dzwoniącego
; FIXME: niestety nie działa jako lokalne dla friend, a callerid nie pozwala na ustawianie ↩
domeny
fromdomain=moja.domena


; plik przeznaczony do umieszczania wpisów postaci:
;  register => [peer?][transport://]user[@domain][:secret[:authuser]]@host[:port][/extension][↩
~expiry]
; typowo postaci:
;  register => login[:password[:authuser]]@host[:port][/extension]
; powodujących rejestrację Asteriska jako perr'a u zewnętrznych operatorów SIP
; umożliwia to zapewnienie wyjścia do ich systemów telefonicznych lub do PSTN
#include <sip-register.conf>


; szablon typowego telefonu i provider'a SIP
[std-phone](!)
        type=friend
; wymagamy rejestracji z użyciem loginu i hasła
host=dynamic

; plik przeznaczony do definiowania telefonów SIPowych
#include <sip-phone.conf>
```

```
; typowe wpisy dla telefonów:
; [USERID](std-phone)
;  ; kontekst w extensions.conf do którego trafią rozmowy przychodzące od "USERID"
;  context = input/USERID
;  ; hasło, które służy do autoryzowania się użytkownika "USERID"
;  secret = haslo
;  ; dane wysyłane jako identyfikacja dzwoniącego
;  callerid = Imie Nazwisko <numer albo identyfikator>


; szablon typowego dostawcy usług SIP
[std-provider](!)
type=peer
; pozwalamy na przyjmowanie połączeń od dostawcy bez autoryzowania
insecure=invite

; plik przeznaczony do definiowania dostawców SIPowych
#include <sip-provider.conf>
; typowe wpisy dla peerów ziwązanych z "register",
; czyli kont u operatorów zewnętrznych z którymi się łączymy:
; [SIP-PROVIDER](std-provider)
;  ; dane umożliwiające logowanie
;  ; (te same co w register - odpowiednio: login, password, host)
;  username=login
;  secret=password
;  host=adres.domenowy.dostawcy
;  ; dane identyfikujące użytkownika u dostawcy
;  fromuser=login
;  fromdomain=domena.konta
;  ; przyjmowane połączenia trafią do podanego w register extension kontekstu:
;  context=input_from/SIP-PROVIDER
; dzwonienie przez operatora realizujemy poprzez:
; Dial(SIP/{$ADDRESS}@user-sipprovider-out)
; uwaga 1: w nazwach kontekstów lepiej nie używać kropek celem łatwego odróżniania od ⤸
domen SIP
; uwaga 2: jeżeli jest kilka kont SIP od tego samego providera nie ma łatwej metody ⤸
kierowanuia
;          ich do różnych kontektstów, należy wtedy kozystać z register z /extension
;          i w wspólnym kontekscie rozrózniać właśnie po tym extension,
;          patrz przykład w: http://www.voip-info.org/wiki/view/Asterisk+config+sip.conf
```

JABBER.CONF

```
[general]
; automatyczna rejestracja i usuwanie ...
;autoregister=yes
;autoprune=yes

; automatyczne akceptowanie subskrypcji
auth_policy=accept


; konto służące do powiadomień o przychodzących połączeniach
[notify]
type=client

; konfiguracja dostępu do serwera XMPP
```

```
serverhost=opcode.eu.org
username=asterisk@opcode.eu.org/asterisk
secret=XXXXXXX
port=5222
usetls=yes

; status i opis
status=available
statusmessage="I am asterisk server"


; konta służące użytkownikowi do komunikacji z voip'em jabberowym (jingle/gtalk)
#include <jabber-users.conf>
; typowe wpisy:
;  [rrp1]
;   type=client
;
;   ; konfiguracja dostępu do serwera XMPP
;   serverhost=opcode.eu.org
;   username=rrp@voip.opcode.eu.org/asterisk
;   secret=XXXXXXXX
;   port=5222
;   usetls=yes
;
;   ; status i opis
;   status=available
;   statusmessage="This is VoIP (jingle/gtalk) only account, real IM account is rrp@opcode.eu.
org"
```

JINGLE.CONF

```
; przykładowa konfiguracja jingle.conf oraz gtalk.conf dla serwera asterisk
; używająca lokalnego serwera jabber'a

[general]
allowguest=yes
bindaddr=::
context=incoming_from_external

; połączenia przychodzące
[guest]
disallow=all
allow=ulaw
context=incoming_from_external

; konfiguracja połączeń dla użytkowników (aby mogli dzwonić poprzez jingle/gtalk)
#include <jingle-outcoming.conf>
;  typowe wpisy:
;  [rrp1]
;   username=rrp@voip.opcode.eu.org
;   disallow=all
;   allow=ulaw
;   ; powiązane konto w jabber-outcoming.conf
;   connection=rrp1
```

DNS

Najczęściej wykorzystywanym serwerem usługi DNS jest bind. Konfiguracja tego programu dzieli się na pliki opisujące opcja związane z działaniem samego serwera i wskazujące na obsługiwane strefy (`/etc/bind/named.conf*`) oraz pliki opisujące poszczególne strefy ich położenie wskazywane jest w ogólnej konfiguracji binda poprzez wpisy typu:

```
zone  "moja.domena" {
        type master;
        file  "/etc/bind/zone/moja.domena.ZONE";
        notify yes;
        allow-update { none; };
};
```

Natomisat same pliki stref są postaci:

```
$ORIGIN moja.domena.
$TTL 86400
@                 IN      SOA     dns.moja.domena.  root.dns.moja.domena. (
                                  2005122902 ;; numer seryjny
                                  7200 ;; sprawdzanie czy zaszly zmiany
                                  7200 ;; ponowna proba kontaktu
                                  604800 ;; dane wygasna po
                                  86400 ;; serwery po drodze beda pamietac przez
                                  )
@                 IN      NS      dns
@                 IN      MX      15      mail
;; GLOWNE
@                 IN      A       XXX.XXX.XXX.XXX ;; UWAGA: dla @ nie można stosować
↪  CNAME gdyż mamy tam już zdefiniowane SOA, MX, NS
                                                  ;;  a CNAME wymaga bycia jedynym
↪  rekordem dla danej subdomeny
dns               IN      A       XXX.XXX.XXX.XXX ;; UWAGA: w myśl RFC 2181 tu nie może
↪  być CNAME bo na ten adres wskazuje NS
mail              IN      A       XXX.XXX.XXX.XXX ;; UWAGA: w myśl RFC 2181 tu nie może
↪  być CNAME bo na ten adres wskazuje MX
im                IN      CNAME   mail.moja.domena.
www               IN      CNAME   dns.moja.domena.
ipv6              IN      AAAA    YYY:YYYY:YYYY::YYYY
;; wpisy SRV dla usług
_xmpp-client._tcp    86400 IN SRV 0 0 5222 im
_xmpp-server._tcp    86400 IN SRV 0 0 5269 im
_sip._udp            86400 IN SRV 0 0 5060 im
_iax._udp            86400 IN SRV 0 0 4569 im
;; wpis dla www (niestety żadna przeglądarka tego nie obsługuje i trzeba dawać @ IN A :-(
_www._tcp            86400 IN SRV 0 0 80   www
;; wpis dla serwera pocztowego - obecnie w zasadzie nie potrzebny (jest MX) ale może
↪  kiedys
_smtp._tcp           86400 IN SRV 0 0 80   mail
;; wpisy SRV z "nioficjalnymi" nazwami potów, tak dla bezpieczeństwa (istotne mogą być
↪  tylko dla jabber)
_http._tcp           86400 IN SRV 0 0 80   www
_mail._tcp           86400 IN SRV 0 0 80   mail
_jabber-client._tcp  86400 IN SRV 0 0 5222 im
_jabber-server._tcp  86400 IN SRV 0 0 5269 im
;; odwrotny DNS dla IP przekierowanych przez CNAME na 23-ip.moja.domena.
;; w odpowiedniej strefie XX.YY.ZZ.in-addr.arpa (u naszego dostawcy) umieszczony musi być
↪  wpis
;;   23 IN CNAME 23-ip.moja.domena.
;; zamiast
```

```
;;   23-ip IN PTR mail.moja.domena.
23-ip              IN       PTR     mail
```

Wspomniane wpisy opcji serwera są następującej postaci:

```
options {
        directory "/var/cache/bind";
        auth-nxdomain no;    # conform to RFC1035
        listen-on-v6 { any; };
        listen-on { any; };
        allow-transfer { any; };
};
```

Warto zwrócić możliwość na wykorzystanie * oznaczającej dowolną nazwę hosta (np. *.moja.domena.pl oznacza wszystkie adresy postaci ciag_znakow_bez-kropki.moja.domena.pl), podobnie możemy uczynić z konfiguracją webserwera (serwer wirtualny z wpisem "ServerAlias *.moja.domena.pl" będzie obsługiwał wszystkie te odwołania (dalej możemy użyć mechanizmów różnicowania położenia katalogu głównego w zależności od tego co jest podane zamiast kropki).

Kolejną rzeczą o której warto wspomnieć jest konfiguracja DNS w bind9 dla IPv6. W `/etc/bind/named.conf.options` dodać należy `listen-on-v6 { any; };` (wewnątrz `options { ... }`). Możemy także konfigurować domeny na adresach IPv6 poprzez dodanie w pliku naszej strefy wpisu z typem AAAA. Odwrotny DNS konfigurujemy podobnie jak dla IPv4 - poprzez utworzenie wpisu dla domeny `{nasz prefix pisany od tyłu po każdej cyfrze szesnastkowej stawiamy kropkę}.ip6.arpa`wraz z wpisami dla hostów - tutaj w rekordach PTR piszemy jako nazwę domeny część odpowiedzialną za adres hosta, również od tyłu i z kropką po każdej cyfrze szesnastkowej.

Niekiedy mogą wystąpić problemy z zatrzymywaniem/restartowaniem binda, może to być spowodowane błędem w kluczy (lub haśle do niego) który autoryzuje komunikację binda i programu go nadzorującego. Należy wtedy przegenerować klucz (`rndc-confgen -r /dev/urandom > /etc/bind/rndc.key`) i dokonać stosowanych zmian w konfiguracji binda (hasło do klucza, ew. jego położenie).

WWW

Do grupy najpopularniejszych usług serwerowych należy zaliczyć protokół HTTP służący do przesyłania stron www i plików. Jednym z najbardziej rozbudowanych i zarazem najpopularniejszych serwerów WWW jest Apache2.

Jego konfiguracja (przynajmniej w Debianie) odbywa się poprzez pliki i katalogi zgomadzone w `/etc/apache2/`. Głównym plikiem konfiguracyjnym jest `apache2.conf`, istotną rolę pełni także `ports.conf` służący do definiowania portów na których ma słuchać serwer (wydaje się też on stosownym miejscem na dyrektywy `NameVirtualHost` wskazujące na jakich adresach obsługiwane mają być hosty wirtualne identyfikowane nazwami). Kolejnymi ważnymi pozycjami są katalogi: `mods-enabled` (zawierający dowiązania do poleceń ładujących i konfigurujących moduły) oraz `sites-enabled` (zawierający konfiguracje udostępnianych stron, vhost'ów).

Listę modułów wraz z opisami można znaleźć w dokumentacji serwera[1721]. Należy pamiętać iż funkcjonalność taka jak: "userdir", "rewriting URL engine", kontrola dostępu (zarówno poprzez dyrektywy Deny, Alow jak i poprzez autoryzację HTTP), automatyczne indeksowanie katalogów, aliasy, obsługa CGI czy proxy zawarta jest w stosowanych modułach.

Przykładowa konfiguracja vhosta może wyglądać:

```
<VirtualHost *:80> # wcześniej musi wystąpić dyrektywa "NameVirtualHost *:80"
        ServerName www.nasza.domena:80
        ServerAlias nasza.domena *.specjalne.nasza.domena
        # *.specjalne.nasza.domena umożliwia współpracę z wpisami dnsowymi postaci:
        #        *.specjalne       IN      A        ....
        # co zapewnia obsługę wszystkich adresów {teskt bez kropki}.test.n17.waw.pl przez
 ↪  ten vhost
        ServerAdmin admin@nasza.domena
        DocumentRoot /var/www/moja-stronka
        # kontrola dostępu oparta na zablokowaniu portu 98 na routerze
        # Order deny,allow
        # Deny from all
        # Allow from 127.0.0.1
```

```
        SSLEngine on
        SSLCertificateFile /etc/ssl/server.crt
        SSLCertificateKeyFile /etc/ssl/server.key
        <Location /status>
                SetHandler server-status
        </Location>
        <Location /info>
                SetHandler server-info
        </Location>
        Alias /mail  "/usr/share/squirrelmail/"
        Alias /mysql "/usr/share/phpmyadmin/"
        Alias /pgsql "/usr/share/phppgadmin/"
        RewriteEngine On
        RewriteRule /webmin https://%{SERVER_NAME}:10000/ [R=301,L]
        RewriteRule /usermin https://%{SERVER_NAME}:20000/ [R=301,L]
</VirtualHost>
```

W głównej konfiguracji apache warto rozważyć ustawienie logowania przez syslog (`ErrorLog syslog:daemon`), dostroić poziom głośności logowania (`LogLevel ...`). Warto wstawić także nastepujące ustawienia i zabezpieczenia zabezpieczenia:

```
UseCanonicalName Off
LogFormat   "%>s %b %T        %h %t %{Host}i \"%r\"         \"%{Referer}i\"
 ↪  \"%{User-Agent}i\""
# jeżeli chemy przekazywać logi do sysloga:
TransferLog |/path-to/apache_syslog
# gdzie apache_syslog jest skryptem o następującej treści:
# /usr/bin/logger -t www -p local0.debug
AddHandler cgi-script .cgi
AddDefaultCharset UTF-8
DirectoryIndex index.html index.cgi index.pl index.php index.xhtml
ExtendedStatus On
<Directory />
        Options All
        AllowOverride All
</Directory>
<Location />
        <Limit CONNECT>
                Order deny,allow
                Deny from all
        </Limit>
</Location>
<Location http://*>
        Order deny,allow
        Deny from all
</Location>
```

A także ustawić zmniejszony timeout w domyślnym vhoście (identyfikowanym nazwą) - pierwszy w kolejności vhost łyka rządzania, których nagłówek Host: nie pasuje do rzadnego innego vhosta - np. `TimeOut 10`.

Niekiedy może zajść potrzeba dostrojenia (np. ustawień `Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec ExecCGI`) pewnych parametrów w konfiguracji udostępniania stron domowych użytkowników poprzez http://serwername/ user w `/etc/apache2/mods-enabled/userdir.conf`. Należy także pamiętać iż ograniczenie ilości danych które możemy przesłać z formularza www do php wynika z konfiguracji tego ostatniego - zmienna `post_max_size = 50M` w `/etc/php5/apache2/php.ini`.

W razie problemów ze startem serwera radzę zwrócić uwagę ściezki gdzie zapisywane są logi - najlepiej podawać od / i upewnić się że katalogi istnieją … . Zamieszam też zestaw prostych skryptów CGI służących do testowania działania tego mechanizmu `cgi-test.tar.gz`

Niekiedy nie potrzeba nam tak rozbudowanego serwera jkim jest Apache - wtedy dobrym wyborem może się okazać mini-httpd (przykładowy konfig[1722]) lub nginx (przykładowe konfigi do ustawienia serwera z obsługą php na porcie 8888 i realizacji proxy do takiego serwera z maszyny bez php - proxy-php.nginx[1723], proxy.nginx[1724] i proxy_params.nginx[1725]).

HTTPS i Apache

Aby uruchomić szyfrowany dostęp do stron WWW (poprzez Apache2) wystarczy uaktywnić moduł SSL, ustawić nasłuch na porcie 443 (standardowy port dla https) oraz skonfigurować vhosta na tym porcie z włączonym szyfrowaniem poprzez umieszczenie w jego konfiguracji:

```
SSLEngine on
SSLCertificateFile /etc/ssl/server.crt
SSLCertificateKeyFile /etc/ssl/server.key
```

Następnie musimy również przygotować odpowiednie certyfikaty - więcej na ten temat w części poświęconej szyfrowaniu z wykorzystaniem certyfikatów x509

Jeżeli wygenerowaliśmy niezależnie certyfikat CA warto w konfiguracji Apache także określić jego położenie:

```
SSLCACertificatePath /etc/ssl/ca/
SSLCACertificateFile /etc/ssl/ca/ca.crt
```

Należy zaznaczyć iż dawniej aby vhosty miały różne certyfikaty SSL należy je uruchamiać na rożnych portach lub adresach IP (jest to ograniczenie wynikłe z samego mechanizmu SSL). Posłużyć się należy dyrektywami: `VirtualHost ip.ip.ip.ip:port` oraz `Listen port` i opcjonalnie - gdy na danym ip i porcie ma być kilka vhostow rozróżnianych nazwami (ale z wspólnym certyfikatem !) `NameVirtualHost ip.ip.ip.ip:port`). Należy też zaznaczyć iż porty możemy użyć w redirectach z http:// (ale z https:// już trudniej - zmiana certyfikatu), najwygodniejsze są adresy IP.

Współczenie rozwiązaniem tego problemu jest wykorzystanie mechanizmu SNI[1726], który pozwala na używanie różnych certyfikatów dla różnych vhostów. Jest on obsługiwany przez wszystkie najpopularniejsze przeglądarki w współczesnych wersjach idomyślnie włączony w Apache.

Kilka słów o znakach z poza ASCII w adresach URI

Zagadnie skład się z dwóch pod-zagadnień:

- kodowania znaków non-ASCII w nazwach domenowych
- kodowania znaków non-ASCII w części określającej ścieżkę, itp

Znaki z poza ASCII w nazwach domenowych są wynalazkiem stosunkowo nowym - przez to nie obciążonym mnogością kodowań narodowych. Zgodnie z dokumentami RFC specyfikującymi to rozwiązanie - Internationalizing Domain Names (RFC3490, RFC3491, RFC3492) człony (od kropki do kropki) takich domen w całości zapisywane są z wykorzystaniem Unicode[1727] kodowanego do postaci zbioru znaków ASCII reprezentowanego w nazwach domenowych z wykorzystaniem punycode[1728]. Standard wspierany jest przez m.in. przez przeglądarkę Firefox 3, ale np. Konqueror 3.5.9 (pomimo oficjalnej zgodności) ma problemy z obsługą tego standardu. W przypadku narzędzi nie potrafiących dekodować IDN wprowadzanego w formie czytelnej dla człowieka (Unicode) konieczne jest zastosowanie programu konwertującego nazwę domeny na punycode - np. ``w3m `idn --quiet http://www.żółw.pl/` ``.

W przypadku znaków z poza ASCII w pozostałych częściach URL sprawa jest bardziej skomplikowana gdyż:

- standard URL[1729] (RFC1738), a później jego uogólnienie URI[1730] (RFC3986) dopuszczają umieszczanie znaków z poza ASCII w adresach URL pod warunkiem zakodowania ich w postaci hexalnej poprzedzonej znakiem % (tzw kodowanie URL)
- jednak żaden z tych standardów ani standard HTTP nie określa kodowania jakie ma być użyte do przypisania wartości numerycznej do znaku - można by zakładać że jest to kodowanie strony na której występuje link (ale takie rozwiązanie rodzi problem z linkami między stronami używającymi różnych kodowań oraz problem ustalenia kodowania przy ręcznym wpisywaniu adresu)
- dopiero standard IRI[1731] (RFC3987) stwierdza jednoznacznie iż takie znaki w URL przed potraktowaniem kodowaniem URL mają zostać zapisane w UTF8. Niestety nadal część aplikacji używa swojego aktualnego kodowania do kodowania znaków nie ASCII w URL zamiast stosować RFC3987.

Dla ścisłości należy zaznaczyć że nie wszystkie znaki ASCII są obsługiwane bezpośrednio (w sposób niekodowany) w URL'ach - niektóre z nich mają znaczenie specjalne w ramach URL, a inne nie są znakami alfanumerycznymi.

Słów kilka o Unicode i kodowaniach

Unicode[1732] sam w sobie przypisuje tylko jednoznaczny kod numeryczny znakom. Nie określa on sposobu zapisu tego kodu numerycznego - za to odpowiadają standardy tupu UCS i UTF:

- UTF-32/UCS-4[1733] - każdemu znakowi odpowiada jedno 32-bitowe słowo
- UCS-2[1734] - każdemu znakowi odpowiada jedno 16-bitowe słowo (nie są reprezentowane wszystkie znaki Unicode!)
- UTF-16[1735] - każdemu znakowi odpowiada co najmniej jedno 16-bitowe słowo, część znaków jest kodowana ciągami dłuższymi
- UTF-8[1736] - każdemu znakowi odpowiada co najmniej jedno 8-bitowe słowo, część znaków jest kodowana ciągami dłuższymi
- UTF-7[1737] - metoda kodowania Unicode dla systemów 7 bitowych
- Punycode[1738] - metoda kodowania Unicode dla systemu nazw domenowych (pierwotnie proponowane były UTF-5 i UTF-6)

Kodowania o długości słowa powyżej 8 bitów (UTF-16, UCS-2, UTF-32/UCS-4) wymagają stosowania znacznika kolejności bajtów BOM[1739] (kodowania te - w odróżnieniu od np. protokołów sieciowych - nie ustalają stałej kolejności bajtów w słowach wielobajtowych). Znacznik ten jest dopuszczany także w UTF-8, jednak jego używanie przy starszych implementacjach jest problematyczne.

Ponadto istnieje kilka metod kodowania znaków 8bitowych (bez względu na to czy są znakami UTF-8 czy jakiegoś kodowania narodowego) do 7-bitowego (lub mniejszego) podzbioru ASCII:

- Quoted-printable[1740]
- Percent-encoding[1741] - zwane także URL-encoding, podobne do Quoted-printable
- Base64[1742]
- Base32[1743]

Poniżej zamieszczam skrypt pokazujący różnice pomiędzy poszczególnymi kodowaniami:

```
NAPIS="ą_ś_ _ź"
echo "ORYGINALNY HEX:"
echo "$NAPIS" | od -t x1
echo -e "\nUTF-32/UCS-4 HEX:"
echo "$NAPIS" | iconv -t utf32 | od -t x4
echo -e "\nUCS2 HEX:"
echo "$NAPIS" | iconv -t ucs2 | od -t x2
echo -e "\nUTF-16 HEX:"
echo "$NAPIS" | iconv -t utf16 | od -t x2
echo -e "\nUTF-8 HEX:"
echo "$NAPIS" | iconv -t utf8 | od -t x1
echo -e "\nUTF-7:"
echo "$NAPIS" | iconv -t utf7
echo -e "\nPunycode:"
echo "$NAPIS" | idn --quiet
echo -e "\nkodowanie Quoted-Printable:"
echo "$NAPIS" | konwert 8bit-qp
echo -e "\nkodowanie URL:"
echo "$NAPIS" | php -r '$stdin=fopen("php://stdin","r"); while($line=fgets($stdin)) {echo
↪  rawurlencode(substr($line,0,strlen($line)-1)) . "\n";}'
echo -e "\nkodowanie BASE64:"
echo "$NAPIS" | mewencode -b
```

Zobacz też: email2text.sh - prosta implementacja konwersji e-maila na zwykły tekst z próbą dekodowania znaków non-ASCII, Comparison of Unicode encodings[1744]

PHP, CGI, suexec - czyli (nie)bezpieczny Apache

Powierzanie dostępu (prawa do odczytu) do poufnych plików (w szczególności plików haseł) użytkownikowi/grupie na prawach której chodzi serwer www (np. apache) jest pomysłem dość kontrowersyjnym, zwłaszcza w systemach wieloużytkownikowych z dostępem do PHP. Problem polega na tym że standardowo PHP działa z tymi samymi prawami co serwer www, zatem jeżeli jakiś plik ukryjemy przed innymi użytkownikami a damy do niego dostęp serwerowi www, to każdy kto może wykonać skrypt PHP przez WWW na tym serwerze będzie miał dostęp do naszego poufnego pliku. Problem ten w szczególności dotyczy też stron zabezpieczanych hasłem przez autoryzację Apache (umieszczaną np. w plikach .htaccess) gdyż nawet gdy plik będzie dostępny dla właściciela i grupy, a będzie maił zabroniony dostęp dla innych użytkowników musi mieć do niego dostęp serwer www (najczęściej przez prawa grupy) a zatem i inni użytkownicy przez PHP.

Pewnym rozwiązaniem tych problemów jest użytkowanie PHP jako CGI (w pierwszej linijce `#!/usr/bin/php-cgi` oraz prawo wykonywania pliku) wraz z korzystaniem z modułu suexec, wykonującego skrypty CGI z prawami innych użytkowników (dla vhostów ustawianych dyrektywą: `SuexecUserGroup nazwa_uzytkownika nazwa_grupy` oraz dla stron użytkowników dostępnych przez tyldą z prawami danego użytkownika). Wykorzystywany jest wtedy program `/usr/lib/apache2/suexec`, który musi mieć właściciela root, grupę www-data (w Debianie) oraz prawa 4750. Oczywiście skrypty cgi są bardziej obciążające dla serwera niż php jako moduł dlatego można stosować równolegle te dwie metody - dla skryptów wymagających poufnych danych odpalamy jako cgi z użytkownikiem mającym dostęp do tych danych (ale nie root ;) dla pozostałych normalnie. Nie rozwiązuje to oczywiście powyższego problemu stron chronionych hasłem (ale stanowi pewne rozwiązanie zabezpieczenia np. haseł do baz danych czy bazy shadow gdy chcemy używać w skryptach autoryzacji PAM (do php można dodać odpowiednie funkcje)), rozwiązanie tamtego problemu wymaga aby nie było w ogóle skryptów php czy cgi chodzących z prawami serwera www.

Na koniec warto zwrócić uwagę że program suexec ma wkompilowaną na stałe w siebie ścieżkę na której musza być uruchamiane z prawami innego użytkownika skrypty (niestety nie jest ona ustawiana żadną dyrektywa konfiguracyjną), jest to ścieżka położenia pliku a nie dojścia do niego więc symlinki z ścieżki do katalogów z root vhostow nie pomogą (w razie wielkich trudności można ustawić przy kompilacji / ale nie jest to zalecane). Warto też zaznaczyć że php przeznaczone do pracy z linii komend w odróżnieniu od wersji kompilowanej dla cgi ma problemy z nagłówkami - nagłówki trzeba wysyłać na samym początku skryptu przez ich wypisywanie (np. echo), a nie przez funkcję header(); wersja cgi domyślnie wysyła też nagłówek określający że jest to html.

Kolejnym problemem jest że skrypty chodzące na prawach serwera www mogą wywołać suexec i odpalić inny skrypt na większych prawach niż posiada serwer www (suexec chodzi z SUID root), zatem zabawa z suexec traci sens gdy pozostaje możliwość odpalenia jakiegokolwiek polecenia z prawamui użytkownika serwera www. Można nawet zaryzykować stwierdzenie że sama obecność suxec w takich systemach stanowi więcej zagrożenia niż pożytku ... .

Szyfrowanie

Przesyłanie danych przez Internet pod względem ich bezpieczeństwa przypomina przesyłanie ich przy pomocy kartek pocztowych - każda osoba (serwer/router) pośrednicząca w przekazywaniu może podejrzeć zawartość, ale zrobić to może także osoba będąca dostatecznie blisko i posiadająca np. lornetkę. Dlatego przy poufnych danych stosuje się ich szyfrowanie. Kolejną kwestia jest łatwość zmiany danych będących w postaci cyfrowej tudzież wyparcia się ich autentyczności/autorstwa - stąd bardzo pokrewna szyfrowaniu technika podpisu cyfrowego[1745]. Zobacz także skrypt eksportujący dane kryptograficzne (i nie tylko) - CommonsConfig-addionals.sh

OpenPGP/GnuPG

Do szyfrowania oraz podpisywania danych możemy skorzystać GnuPG[1746].

Aby **wygenerować** klucze uruchamiamy `gpg --gen-key`, aby dodać dodatkowy identyfikator do naszego klucza (przydatne gdy chcemy nim podpisywać np. maile wysyłane z różnych adresów uruchamiamy `gpg --edit-key podany.przy.generacji@email.identyfikujacy.klucz` i w uzyskanej linii poleceń wydajemy `adduid`.

Aby **wyeksportować** nasz klucz publiczny używamy komendy `gpg --armour --output publiczny.gpg --export podany.przy.generacji@email.identyfikujacy.klucz`, aby wyeksportować klucz prywatny używamy opcji `gpg --list-secret-keys`. Możemy także uzyskać bardzo streszczony klucz publiczny (pozbawiony wszelkich podpisów innych osób) - w tym celu należy wygenerować go na podstawie klucza prywatnego w następujący sposób:

```
mkdir gpg-export-tmp-dir
chmod 700 gpg-export-tmp-dir
gpg -a --export-secret-keys > gpg-export-tmp-dir/sec.asc
gpg --homedir gpg-export-tmp-dir --import gpg-export-tmp-dir/sec.asc
gpg --homedir gpg-export-tmp-dir -a --export > public.asc
rm -fr gpg-export-tmp-dir
```

Aby **zaimportować** czyjś klucz `gpg --import publiczny.gpg`.

Aby **zaszyfrować** dane wysyłane do kogoś jego kluczem publicznym: `gpg --output plik_wyjsciowy --encrypt --recipient identyfikator@email.odbiorcyplik_wejsciowy`, aby **odszyfrować** odebrane dane szyfrowane naszym kluczem publicznym: `gpg --output plik_wyjsciowy --decrypt plik_wejsciowy`. Gdy dysponujemy zestawem klucza publicznego i prywatnego w formacie binarnym możemy to uczynić tak-

że bez importowania tych kluczy: `gpg --keyring klucz_publiczny --secret-keyring klucz_prywatny --decrypt wiadomosc_do_odszyfrowania`. Jeżeli chcemy odszyfrować dane mając klucz prywatny w postaci ASCII nie obędzie się bez jego zaimportowania, można to jednak zrobić do tymczasowego pliku:

```
mkdir gpg-decrypt-tmp-dir
chmod 700 gpg-decrypt-tmp-dir
gpg --homedir gpg-decrypt-tmp-dir --import secret.asc
gpg --homedir gpg-decrypt-tmp-dir --decrypt zaszyfrowany.txt > odszyfrowany.txt
rm -fr gpg-decrypt-tmp-dir
```

Aby **wygenerować podpis** do wiadomości: `gpg --output wygenerowany_podpis.sig --detach-sig plik_wejsciowy`.

Zobacz też: Architektura klucza publicznego w Gnu Privacy Guard (GPG)[1747] oraz klucze ssh, VPN.

Certyfikaty x509

Do szyfrowania i podpisywania możemy korzystać także z z certyfikatów takich jak wykorzystywane w połączeniach SSL. Istnieje kilka dróg generowania certyfikatów i kluczy tego typu. Możemy to zrobić np. w następujący sposób: `openssl req -new -x509 -days 4095 -newkey rsa:2048 -keyout ca_key.pem -out ca_cert.pem`, wygenerowany zostanie certyfikat podpisany przez samego siebie (opcja `-x509`), jeżeli dodana byłaby opcja `-nodes` klucz prywatny nie byłby zabezpieczony hasłem (jest to przydatne przy generowaniu certyfikatów dla serwerów - szyfrowanie www, poczty itp.).

Wygenerowany w sposób opisany powyżej zestaw certyfikatu i klucza możemy używać od razu do szyfrowania bądź do podpisywania próśb o certyfikat generowanych komendą: `openssl req -new -keyout server.key -out server.key` (UWAGA: w przypadku certyfikatów dla serwerów w "COMMON NAME" nazwa domeny dla której jest certyfikat, można użyć *.domena aby pasował do wszystkich hostów w domenie). Prośbę taką możemy przekazać do podpisu jakiemuś prawdziwemu CA lub podpisać sobie sami (wygenerowanym wcześniej kluczem): `openssl x509 -req -out server.crt -in server.key -CA ca_cert.pem -CAkey ca_key.pem -CAcreateserial -days 365`. Na koniec możemy usunąć hasło z klucza prywatnego (przydatne dla serwerów): `openssl rsa -in server.key -out server.key`.

Na potrzeby wielu programów konieczne będzie wyeksportowanie zestawu klucza i certyfikatu do formatu PKCS#12 - `openssl pkcs12 -export -in cert1_cert.pem -inkey cert1_key.pem -out cert1.p12`. Klucz taki możemy zaimportować np. do **gpgsm** - `gpgsm --import cert1.p12` (wcześniej odpalamy gpg-agent –daemon i eksportujemy tak jak sugeruje GPG_AGENT_INFO). Pozwoli to na włączenie funkcjonalności S/MIME[1748] w **kmail**'u (wymagane pakiety gpgsm i kleopatra), w tym celu należy ponadto zaimportować certyfikaty CA - `gpgsm --import /etc/ssl/certs/ca-certificates.crt` i zakwalifikować wszystkie te (nasz i CA) certyfikaty jako zaufane - `gpgsm -k | grep fingerprint | while read tmp fpr; do echo "$fpr S relax"; done > ~/.gnupg/trustlist.txt`

Problem wielu domen

Znanym problemem związanym z certyfikatami SSL jest problem obsLugi wielu domen na jednym zestawie ip-port. Pierwotnie protokół przywiązał do takiego zestawy jeden cetryfikat co uniemożliwiaLo stawiania name based virtual host z ssl-em. Obecnie stosowane jest kilka rozwiązań tego problemu (jednak wszystkie miewają jakieś ograniczenia ze względu na kompatybilność z starszym oprogramowaniem):

— wystawianie certyfikatów dla *.domena
— stosowanie kilku CN w jednym certyfikacie
— stosowanie dodatkowej informacji "DNS" w certyfikacie
— mechanizm Server Name Indication[1749] pozwalający na podanie nazwy domenowej w trakcie negocjacji połączenia

Przykładowa metoda generacji samopodpisanego certyfikatu mającego kilka CN i wpisów "DNS":

```
openssl genrsa -out server.key 1024
openssl req -new -config server.cfg -key server.key -out server.csr
# uzyskane żądanie certyfikatu możemy obejrzeć przy pomocy
# openssl req -in server.csr -text -noout
# samopodpisanie certyfikatu
openssl x509 -req -in server.csr -signkey server.key -extfile server.cfg -extensions
↪ v3_req -days 4095 -text -out server.crt
```

Wykorzystywany plik konfiguracyjny `server.cfg` ma postać:

```
[ req ]
default_bits            = 1024
distinguished_name      = req_distinguished_name
req_extensions          = v3_req
prompt                  = no
[ v3_req ]
basicConstraints        = CA:FALSE
keyUsage                = nonRepudiation, digitalSignature, keyEncipherment
subjectAltName          = @alt_names
[ req_distinguished_name ]
C=PL
L=miasto
O=organizacja
OU=jednostka
emailAddress=admin@example.org
[ alt_names ]
DNS.0 = sub.domena1.tld
DNS.1 = domena2.tld
[ req_distinguished_name ]
0.CN=sub.domena1.tld
1.CN=domena2.tld
```

Router - przekazywanie pakietów, DHCP itp

Skrypty do routingu i podziału pasma

Prezentuję tutaj kilka skryptów dokonujących konfiguracji takiej usługi w systemie Linux: routing.sh, routing.conf i routing_fun.sh odpowiedzialne za routing IPv4 i IPv6. Skrypty te wymagają pela wraz z NetAddr::IP::Util (pakiet libnetaddr-ip-perl w wersji conajmniej 4.007), a także pakietów iptables, iproute oraz jądra z obsługą iptables, ipv6, htb i innych (zalecana jest obsługa imq w trybie AB).

Pokazane w przykładowym pliku konfiguracyjnym markowanie pakietów (ustawianie numerycznych znaczników dostępnych tylko w ramach hosta routującego) możemy użyć nie tylko jak tam pokazano do `tc filter` czy `iptables` ale także do kierowania ruchu do różnych tablic routingu. Możemy wykorzystać do tego polecenie `ip rule` poprzez wykorzystanie selektora `fwmark` i celu `table`. Innymi ciekawymi selektorami są: `tos` (wybór w oparciu o pole TOS), `from` lub `to` (adres źródłowy lub docelowy), `iif` (interfejs z którego przyszedł pakiet), a z celów `nat`, `reject`. Różne tabele routingu możemy tworzyć poprzez `ip route table NAZWA_TABELI`, przy pomocy tego narzędzia możemy tworzyć zaawansowane mechanizmy routingu (routing lokalny, multicastowy, rozgłoszeniowy, ...). Z innych mniej znanych a ciekawych opcji komendy `ip` warto wspomnieć o: `maddress`, `link`, `neighbour`, `tunel`. Więcej szczegółów w `man ip`.

Ponadto prezentuję skrypty generujące i prezentujące statystyki obciążenia łącza: traffic_stats.sh i traffic_stats_show.sh oraz skrypt startowy traffic_stats.init.sh.

routing.sh

```
#!/bin/bash

# skrypt konfigurujący routing i podział pasma
# pozwala na wykorzystanie NAT, podziału w oparciu o kolejkę priorytową oraz htb
#
# wydaje się że skrypt można dość łatwo rozszerzyć na przypadek dwuch łącza poprzez
# powielenie całości kodu kodu htb, należy oczywiście zadbać o odpowiednią ⤸
konfigurację
# routingu taką jak opisana w rozdziale 4.2 HOWTO Kształtowanie Ruchu i Zaawansowany ⤸
Routing
# http://linuxreviews.org/howtos/networking/lartc/pl/x271.html
```

```
#
# tunel ipv6 może potencjalnie wywoływać problem polegający na podwójnym liczeniu jego
# ruchu (jako ruchu wewnątrz tunelowego jak i ruchu ipv4 tunelu), jego rozwiązaniem
# byłoby wyfiltrowanie ruchu ipv4 tunelu (puszczenie go w klasie nielimitowanej)
#
# ogólnie należy zaznaczyć iż tunele i kolejkowanie to "chore" pomysly (lepiej natywne
łącze i na tyle szerokie
# aby nie trzebabyło kolejkować, a priorytetyzacje ruchu realizować w oparciu o vlany)



# włączanie zbioru przydatnych funkcji
. /usr/local/lib/routing_fun.sh

# włączanie pliku konfiguracyjnego
. /etc/routing.conf

#################
##      IPv6      ##
#################

start_ipv6() {
 # będziemy używać tuneli (zwykłych)
 modprobe ip6_tunnel

# nie korzystamy natomiast z tuneli typu sit
 rmmod sit

# usuwamy routing z interfejsu zewnętrznego
 eval ip -6 route del ff00::/8 dev ${name[$ext_conf]} $QUITE
 eval ip -6 route del fe80::/64 dev ${name[$ext_conf]} $QUITE

# nie korzystamy z adresów link-scope (experimental ;-) )
 for dev in `ip link sh | grep mtu | cut -f2 -d' ' | tr -d ':' | cut -f1 -d'@'`; do
  if [ "$dev" != "${name[$int_conf]}" ]; then # radvd wymaga adresu link-scope
   ip=`ip addr sh dev $dev | grep 'inet6 .* scope link' | awk '{print $2}'`;
   if [ "$ip" != "" ]; then ip addr del $ip dev $dev; fi;
  fi
 done;

for X in `seq 0 $(( ${#mode[@]} - 1)) `; do
  if [ "${mode[$X]}" = "sit" ]; then
   # konfigurujemy tunel
   ip tunnel add ${name[$X]} mode sit remote ${remote[$X]} local ${local[$X]}
   ip link set ${name[$X]} up
   ip addr add ${base_ipv6[$X]}/${base_maskv6[$X]} dev ${name[$X]}
   # uwaga tunel moze wymagac zmniejszenia mtu na komputerach klienckich:
   #    ifconfig eth0 mtu 1280
   # celem poprawnego dzialania niektorych stron WWW itp. - np. www.ripe.net

   # usuwamy routing locallink z interfejsu tunelu
   eval ip -6 route del fe80::/64 dev ${name[$X]} $QUITE

   # dodajemy trasę domyślną dla tunelu
   ip -6 route add ::/0 metric 256 dev ${name[$X]}
  elif [ "${mode[$X]}" = "int" ]; then
   # usuwamy trasę domyślną z interfejsów (poza tunelami)
   eval ip -6 route del ::/0 dev ${name[$X]} $QUITE
  fi;
 done;
```

```
# dodajemy routing na interfejsie wewnętrznym
 if [ "${base_ipv6[$int_conf]}" != "" ]; then
  ip -6 route add `ipv6_and_mask2network ${base_ipv6[$int_conf]} ${base_maskv6[$int_conf]}`/${
  base_maskv6[$int_conf]} \
   metric 64 dev ${name[$int_conf]}
 fi
}

stop_ipv6() {
 for tun in `ip tunnel sh | awk 'BEGIN{ FS="[: \t]+"} $2=="ipv6/ip" && $1!="sit0" {print $1}
 '`; do
  ip tunnel del $tun
 done
 if [ "${base_ipv6[$int_conf]}" != "" ]; then
  ip -6 route del `ipv6_and_mask2network ${base_ipv6[$int_conf]} ${base_maskv6[$int_conf]}`/${
  base_maskv6[$int_conf]} \
   dev ${name[$int_conf]}
 fi
}


##################
##      ROUTING      ##
##################

start_routing() {
 # włączenie IP forwarding
 echo "1" > /proc/sys/net/ipv4/conf/all/forwarding
echo "1" > /proc/sys/net/ipv6/conf/all/forwarding

# przekazywanie multicastu IPv4
 route add -net 224.0.0.0 netmask 240.0.0.0 dev ${name[$int_conf]}
}

stop_routing() {
 # wyłączenie przekazywania multicastu IPv4
 route del -net 224.0.0.0 netmask 240.0.0.0 dev ${name[$int_conf]}

# wyłączenie IP forwarding
 echo "0" > /proc/sys/net/ipv4/conf/all/forwarding
echo "0" > /proc/sys/net/ipv6/conf/all/forwarding
}


###################
##      IPTABLES      ##
###################

start_ip6tables() {
 # blokady portów
 for port in $block_ports; do
  ip6tables -A INPUT   -i ${name[$extv6_conf]} -p tcp --dport $port -j REJECT
  ip6tables -A FORWARD -i ${name[$extv6_conf]} -p tcp --dport $port -j REJECT
done;

# zezwalamy na przekazywanie pakietow z naszej sieci na zewnatrz i z zewnatrz do naszej
 sieci
 # (przez odpowiednie uzadzenia) - określone w konfiguracji adresy IP
 for ip in `echo "$forwarding_ip" | cut -s -f2 -d'|' | tr ' ' '\n'`; do
```

```
  ip6tables -A FORWARD -i ${name[$int_conf]} -s $ip -o ${name[$extv6_conf]} -d 0.0.0.0/0 -j ⤸
  ACCEPT
  ip6tables -A FORWARD -i ${name[$extv6_conf]} -s 0.0.0.0/0 -o ${name[$int_conf]} -d $ip -j ⤸
  ACCEPT
done
 ip6tables -A FORWARD -i ${name[$int_conf]} -o ${name[$extv6_conf]} -j REJECT
ip6tables -A FORWARD -i ${name[$extv6_conf]} -o ${name[$int_conf]} -j REJECT

# logowanie połączeń przychodzących z sieci wewnętrznej (do serwera i sieci ⤸
 zewnętrznych)
 ip6tables -A PREROUTING  -t mangle -i ${name[$int_conf]} -m state --state NEW \
  -j LOG --log-level debug --log-prefix "FROM-INTv6: "

# logowanie połączeń przychodzących z sieci zewnętrznej (do serwera i sieci ⤸
 wewnętrznych)
 ip6tables -A PREROUTING  -t mangle -i ${name[$extv6_conf]} -m state --state NEW \
  -j LOG --log-level debug --log-prefix "FROM-EXTv6: "

# logowanie połączeń wychodzących do sieci zewnętrznych
 #ip6tables -A POSTROUTING -t mangle -o ${name[$extv6_conf]} -m state --state NEW \
  -j LOG --log-level debug --log-prefix "TO-EXTv6: "
ip6tables -A OUTPUT -o ${name[$extv6_conf]} -m state --state NEW \
  -j LOG --log-level debug --log-prefix "TO-EXTv6: "
}

stop_ip6tables() {
 # wykasowanie dotychczasowych regół + wszystko czego nie zabronimy puszczamy
 ip6tables -F -t filter
ip6tables -F -t mangle
ip6tables -F -t raw
ip6tables -P INPUT ACCEPT
ip6tables -P OUTPUT ACCEPT
ip6tables -P FORWARD ACCEPT
}

start_iptables() {
 # blokady portów
 for port in $block_ports; do
  iptables -A INPUT   -i ${name[$extv_conf]} -p tcp --dport $port -j REJECT
  iptables -A FORWARD -i ${name[$extv_conf]} -p tcp --dport $port -j REJECT
done;

# kontrola ToS
 iptables -A PREROUTING -s ! $voip_host_ipv4 -d ! $voip_host_ipv4 -t mangle -j TOS --set-tos ⤸
0x00

# zezwalamy na przekazywanie pakietow z naszej sieci na zewnatrz i z zewnatrz do naszej ⤸
 sieci
 # (przez odpowiednie uzadzenia) - multicast oraz określone w konfiguracji adresy IP
 for ip in "224.0.0.0/4" `echo "$forwarding_ip" | cut -f1 -d'|' | tr ' ' '\n'`; do
  iptables -A FORWARD -i ${name[$int_conf]} -s $ip -o ${name[$ext_conf]} -d 0.0.0.0/0 -j ⤸
  ACCEPT
  iptables -A FORWARD -i ${name[$ext_conf]} -s 0.0.0.0/0 -o ${name[$int_conf]} -d $ip -j ⤸
  ACCEPT
done
 # pozostały forwarding blokujemy
 iptables -A FORWARD -i ${name[$int_conf]} -o ${name[$ext_conf]} -j REJECT
iptables -A FORWARD -i ${name[$ext_conf]} -o ${name[$int_conf]} -j REJECT

# logowanie połączeń przychodzących z sieci wewnętrznej (do serwera i sieci ⤸
```

```
zewnętrznych)
 iptables -A PREROUTING  -t mangle -i ${name[$int_conf]} -m state --state NEW \
  -j LOG --log-level debug --log-prefix "FROM-INT: "

# logowanie połączeń przychodzących z sieci zewnętrznej (do serwera i sieci ⤸
 wewnętrznych)
 iptables -A PREROUTING  -t mangle -i ${name[$ext_conf]} -m state --state NEW \
  -j LOG --log-level debug --log-prefix "FROM-EXT: "

# logowanie połączeń wychodzących do sieci zewnętrznych
 #iptables -A POSTROUTING -t mangle -o ${name[$ext_conf]} -m state --state NEW \
  -j LOG --log-level debug --log-prefix "TO-EXT: "
iptables -A OUTPUT -o ${name[$ext_conf]} -m state --state NEW -j LOG --log-level debug --log-⤸
prefix "TO-EXT: "

# generowane powyżej logi można wykorzystać np. do wychwycenia wirusów rozsyłających ⤸
 spam
 # (duża ilość połączeń z zewn. SMTP) przy pomocy koemndy:
 # grep ' DPT=25 ' /var/log/kernel-debug.log | grep ' OUT=eth1 ' | grep ' FROM-INT: ' |
 # awk 'BEGIN{FS="[= ]+"} {tablica[$11]++} END{for (INDEX in tablica) printf("%s %s\n", INDEX,⤸
 tablica[INDEX])}' |
 # sort -n -k2

# SNAT lub MASQUERADE dla adresow z $nat_network
 if [ "$nat_mode" = "SNAT" ]; then
  iptables -A POSTROUTING -t nat -o ${name[$ext_conf]} -s $nat_network -d 0.0.0.0/0 \
   -j SNAT --to-source ${base_ipv4[$ext_conf]}
 elif [ "$nat_mode" = "MASQUERADE" ]; then
  iptables -A POSTROUTING -t nat -o ${name[$ext_conf]} -s $nat_network -d 0.0.0.0/0 -j ⤸
  MASQUERADE
fi
}

stop_iptables() {
 # wykasowanie dotychczasowych regół + wszystko czego nie zabronimy puszczamy
 iptables -F -t filter
iptables -F -t nat
iptables -F -t mangle
iptables -F -t raw
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
}


##################################
##       TrafficControl - UPLOAD       ##
##################################

start_tc_up() {
 if $use_imq; then
  # ładujemy moduł imq
  modprobe imq

  # kirujemy ruch do urządzenia IMQ
  if [ "$ext_conf" != ""  -a  "${name[$ext_conf]}" != "" ]; then
   iptables -t mangle -A FORWARD -i ${name[$int_conf]} -o ${name[$ext_conf]} -j IMQ --todev 0
   iptables -t mangle -A OUTPUT -o ${name[$ext_conf]} -j IMQ --todev 0
  fi
  if [ "$extv6_conf" != ""  -a  "${name[$extv6_conf]}" != "" ]; then
```

```
   ip6tables -t mangle -A FORWARD -i ${name[$int_conf]} -o ${name[$extv6_conf]} -j IMQ --⤸
   todev 0
   ip6tables -t mangle -A OUTPUT -o ${name[$extv6_conf]} -j IMQ --todev 0
  fi

  # ustawiamy że tc chodzi na imq0
  tc_up_dev=imq0
  # włączamy urządzenie imq
  ip link set $tc_up_dev up

  set_mtu="mtu 16500"
  # zapobiega segmentowaniu pakietów UDP, co prowadzić by mogło do nietrzymania kolejek w ⤸
  IMQ,
  # (wystąpienie tego problemu można zaobserować po niezerowej wartości ”giants” w tc -s ⤸
  class show dev imqX)
 else
  set_mtu=""
  tc_up_dev=${name[$ext_conf]}
 fi


## kolejeki podstawowe
 if $use_prio_queue; then
  # tworzymy kolejke glowna PRIO dla naszego interfejsu wewnetrznego (identyfikowana przez 5:⤸
  0)
  tc qdisc add dev $tc_up_dev root handle 5:0 prio bands 2 priomap 1 1 1 1 1 1 1 1 1 0 1 1 ⤸
  1 1 1
  # domyślnym jest :2 ... do :1 wpuszczamy z ToS == 0x14 oraz na podstawie poniższyzych ⤸
  filtrów
  # tos jest ograniczany do voip_host poprzez iptables

  # sprawiedliwy dostęp do kabla wielu jednoczesnych połączeń w kolejce priorytetowej
  # (identyfikowana przez 5:1)
  tc qdisc add dev $tc_up_dev parent 5:1 sfq perturb 10

  # kolejka htb dla połączeń nie priorytetowych, klasa na której jest tworzona
  # jest wskazana w ”tc qdisc add ... prio ...” (identyfikowana przez 5:2)
  htb_handle="6"
  tc qdisc add dev $tc_up_dev parent 5:2 handle $htb_handle:0 htb default 7
 else
  # tworzymy kolejke glowna HTB dla naszego interfejsu wewnetrznego (identyfikowana przez 1:0)
  htb_handle="5"
  tc qdisc add dev $tc_up_dev root handle $htb_handle:0 htb default 7
 fi

# tworzymy klase HTB odpowiadajaca predkosci naszego łącza
 # (identyfikowana przez $htb_handle:4 bo :1,:2 jest zarezerwowane - priorytety)
 tc class add dev $tc_up_dev parent $htb_handle:0 classid $htb_handle:4 htb \
  rate $full_speed_up ceil $full_speed_up $set_mtu

# gdy HTB podstawową kolejką tworzymy w niej klasę ruchu priorytetowego
 # (identyfikowana przez 5:1)
 if [ $htb_handle = 5 ]; then
  # pożycza ona pasmo od innych (borrow), nie udostępnia pasma (isolated)
  tc class add dev $tc_up_dev parent $htb_handle:4 classid $htb_handle:1 htb \
   rate $pri_speed_up ceil $full_speed_up prio 1
  # sprawiedliwy dostęp do kabla wielu jednoczesnych połączeń
  tc qdisc add dev $tc_up_dev parent $htb_handle:1 sfq perturb 10
 fi
```

```
# tworzymy klase HTB dla ruchu generowanego przez użytkowników (identyfikowana przez ⤸
 $htb_handle:2)
 tc class add dev $tc_up_dev parent $htb_handle:4 classid $htb_handle:2 htb \
  rate $speed_up ceil $speed_up burst 12k quantum 2000


# filtry kolejeki ruchu priorytetowego (identyfikowanej przez 5:1)
 if [ "$voip_host_ipv4" != "" ]; then
  # filtracja w oparciu o port - IAX2 na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_up_dev protocol ip   $TC_PREF parent 5:0 u32 \
   match ip  sport 4569 0xffff match ip  src $voip_host_ipv4 flowid 5:1
  tc_pref_inc 3; tc filter add dev $tc_up_dev protocol ip   $TC_PREF parent 5:0 u32 \
   match ip  dport 4569 0xffff match ip  src $voip_host_ipv4 flowid 5:1

  # filtracja voipu (w oparciu o ToS) na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_up_dev protocol ip   $TC_PREF parent 5:0 u32 \
   match ip  tos 0x14 0xff match ip  src $voip_host_ipv4 flowid 5:1
 fi
 if [ "$voip_host_ipv6" != "" ]; then
  # filtracja w oparciu o port - IAX2 na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_up_dev protocol ipv6 $TC_PREF parent 5:0 u32 \
   match ip6 sport 4569 0xffff match ip6 src $voip_host_ipv6 flowid 5:1
  tc_pref_inc 3; tc filter add dev $tc_up_dev protocol ipv6 $TC_PREF parent 5:0 u32 \
   match ip6 dport 4569 0xffff match ip6 src $voip_host_ipv6 flowid 5:1

  # filtracja voipu (w oparciu o ToS) na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_up_dev protocol ipv6 $TC_PREF parent 5:0 u32 \
   match ip  tos 0x14 0xff match ip6 src $voip_host_ipv6 flowid 5:1
 fi


# klasa ruchu niepriorytetowego - serwery (identyfikowana $htb_handle:8)
 tc class add dev $tc_up_dev parent $htb_handle:2 classid $htb_handle:8 htb \
  rate $serw_speed_up ceil $speed_up burst 12k quantum 2000 prio 3
 # uzyskanie adresów sieci serwerowej - konfiguracja z interfejsu wewnetrznego NIE jest ⤸
 pomyłką
 # sieć serwerowa siedzi wewnątrz i jest podstawową siecią wewnętrzną
 net_addr=`ip_and_mask2network ${base_ipv4[$int_conf]} ${base_maskv4[$int_conf]}`
tc_pref_inc 5
 tc filter add dev $tc_up_dev protocol ip $TC_PREF parent $htb_handle:0 u32 \
  match ip src $net_addr/${base_maskv4[$int_conf]} flowid $htb_handle:8
 tc qdisc add dev $tc_up_dev parent $htb_handle:8 sfq perturb 10


# klasa ruchu niepriorytetowego - reszta (identyfikowana $htb_handle:7)
 # jest ona wskazana w parametrze "default" w "tc qdisc add ... htb"
 tc class add dev $tc_up_dev parent $htb_handle:2 classid $htb_handle:7 htb \
  rate $lan_speed_up ceil $speed_up burst 12k quantum 2000 prio 5
 tc qdisc add dev $tc_up_dev parent $htb_handle:7 sfq perturb 10
}

stop_tc_up() {
 # kasujemy poprzednie ustawienia kolejek, filtrów, ...
 eval tc qdisc del root dev ${name[$int_conf]} $QUITE
 eval tc qdisc del root dev imq0 $QUITE
}


#####################################
##      TrafficControl - DOWNLOAD      ##
```

```
####################################

start_tc_down() {
 if $use_imq; then
  # ładujemy moduł imq
  modprobe imq

  # kirujemy ruch do urządzenia IMQ
  if [ "$ext_conf" != ""  -a  "${name[$ext_conf]}" != "" ]; then
   iptables -t mangle -A FORWARD -i ${name[$ext_conf]} -o ${name[$int_conf]} -j IMQ --todev 1
   iptables -t mangle -A INPUT -i ${name[$ext_conf]} -j IMQ --todev 1
  fi
  if [ "$extv6_conf" != ""  -a  "${name[$extv6_conf]}" != "" ]; then
   ip6tables -t mangle -A FORWARD -i ${name[$extv6_conf]} -o ${name[$int_conf]} -j IMQ --⤸
   todev 1
   ip6tables -t mangle -A INPUT -i ${name[$extv6_conf]} -j IMQ --todev 1
  fi

  # ustawiamy że tc chodzi na imq1
  tc_down_dev=imq1
  # włączamy urządzenie imq
  ip link set $tc_down_dev up

  set_mtu="mtu 16500"
else
  set_mtu=""
  tc_down_dev=${name[$int_conf]}
 fi


## kolejeki podstawowe
 if $use_prio_queue; then
  # tworzymy kolejke glowna PRIO dla naszego interfejsu wewnetrznego (identyfikowana przez 1:⤸
  0)
  tc qdisc add dev $tc_down_dev root handle 1:0 prio bands 2 priomap 1 1 1 1 1 1 1 1 1 1 1 1 ⤸
  1 1 1 1
  # domyślnym jest :2 ... do :1 na podstawie poniższych filtrów

  # sprawiedliwy dostęp do kabla wielu jednoczesnych połączeń w kolejce priorytetowej
  # (identyfikowana przez 1:1)
  tc qdisc add dev $tc_down_dev parent 1:1 sfq perturb 10

  # kolejka htb dla połączeń nie priorytetowych, klasa na której jest tworzona
  # jest wskazana w "tc qdisc add ... prio ..." (identyfikowana przez 1:2)
  htb_handle="2"
  tc qdisc add dev $tc_down_dev parent 1:2 handle $htb_handle:0 htb default 7
 else
  # tworzymy kolejke glowna HTB dla naszego interfejsu wewnetrznego (identyfikowana przez 1:0)
  htb_handle="1"
  tc qdisc add dev $tc_down_dev root handle $htb_handle:0 htb default 7
 fi

# tworzymy klase HTB odpowiadajaca predkosci naszego interfejsu wewnetrznego (identyfikowana ⤸
 przez 2$htb_handle:6)
 tc class add dev $tc_down_dev parent $htb_handle:0 classid $htb_handle:6 htb \
  rate ${interface_speed[$int_conf]} ceil ${interface_speed[$int_conf]} burst 12k quantum ⤸
  2000 $set_mtu

# tworzymy klase HTB dla ruchu wewnętrznego (identyfikowana przez $htb_handle:5)
 tc class add dev $tc_down_dev parent $htb_handle:6 classid $htb_handle:5 htb \
```

```
  rate $int_speed_down ceil $int_speed_down burst 12k quantum 2000
 # sprawiedliwy dostęp do kabla wielu jednoczesnych połączeń
 tc qdisc add dev $tc_down_dev parent $htb_handle:5 sfq perturb 10

# tworzymy klase HTB odpowiadajaca predkosci naszego łącza
 # (identyfikowana przez $htb_handle:4 bo :1,:2 jest zarezerwowane - priorytety)
 tc class add dev $tc_down_dev parent $htb_handle:0 classid $htb_handle:4 htb \
  rate $full_speed_down ceil $full_speed_down

# gdy HTB podstawową kolejką tworzymy w niej klasę ruchu priorytetowego
 # (identyfikowana przez 1:1)
 if [ $htb_handle = 1 ]; then
  # pożycza ona pasmo od innych (borrow), nie udostępnia pasma (isolated)
  tc class add dev $tc_down_dev parent $htb_handle:4 classid $htb_handle:1 htb \
   rate $pri_speed_down ceil $full_speed_down burst 12k quantum 2000 prio 1
  # sprawiedliwy dostęp do kabla wielu jednoczesnych połączeń
  tc qdisc add dev $tc_down_dev parent $htb_handle:1 sfq perturb 10
 fi

# tworzymy klase HTB dla ruchu generowanego przez użytkowników (identyfikowana przez ⤸
$htb_handle:2)
 tc class add dev $tc_down_dev parent $htb_handle:4 classid $htb_handle:2 htb \
  rate $speed_down ceil $speed_down burst 12k quantum 2000


# filtry kolejki ruchu wewnętrznego
 for ip in ${base_ipv4[@]} ${other_ipv4[@]}; do
  tc_pref_inc 1
  tc filter add dev $tc_down_dev protocol ip   $TC_PREF parent $htb_handle:0 u32 \
   match ip  src $ip flowid $htb_handle:5
 done
 for ip in ${base_ipv6[@]} ${other_ipv6[@]}; do
  tc_pref_inc 1
  tc filter add dev $tc_down_dev protocol ipv6 $TC_PREF parent $htb_handle:0 u32 \
   match ip6 src $ip flowid $htb_handle:5
 done

# filtry kolejeki ruchu priorytetowego
 if [ "$voip_host_ipv4" != "" ]; then
  # filtracja w oparciu o port - IAX2 na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_down_dev protocol ip   $TC_PREF parent 1:0 u32 \
   match ip  sport 4569 0xffff match ip  dst $voip_host_ipv4 flowid 1:1
  tc_pref_inc 3; tc filter add dev $tc_down_dev protocol ip   $TC_PREF parent 1:0 u32 \
   match ip  dport 4569 0xffff match ip  dst $voip_host_ipv4 flowid 1:1

  # filtracja voipu (w oparciu o ToS) na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_down_dev protocol ip   $TC_PREF parent 1:0 u32 \
   match ip  tos 0x14 0xff match ip  dst $voip_host_ipv4 flowid 1:1
 fi
 if [ "$voip_host_ipv6" != "" ]; then
  # filtracja w oparciu o port - IAX2 na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_down_dev protocol ipv6 $TC_PREF parent 1:0 u32 \
   match ip6 sport 4569 0xffff match ip6 dst $voip_host_ipv6 flowid 1:1
  tc_pref_inc 3; tc filter add dev $tc_down_dev protocol ipv6 $TC_PREF parent 1:0 u32 \
   match ip6 dport 4569 0xffff match ip6 dst $voip_host_ipv6 flowid 1:1

  # filtracja voipu (w oparciu o ToS) na wybranym hoscie
  tc_pref_inc 3; tc filter add dev $tc_down_dev protocol ipv6 $TC_PREF parent 1:0 u32 \
   match ip  tos 0x14 0xff match ip6 dst $voip_host_ipv6 flowid 1:1
 fi
```

```
# ustawienia uzytkownikow
 numerek=9
 echo "$forwarding_ip" | while read line; do
  curr_speed=`echo "$line" | cut -s -f3 -d'|'`
  if [ "$curr_speed" = "" ]; then
   curr_speed=$user_speed_down
  fi
  curr_speed_max=`echo "$line" | cut -s -f4 -d'|'`
  if [ "$curr_speed_max" = "" ]; then
   curr_speed_max=$speed_down
  fi
  #zwiekszamy numerek klasy
  numerek=$(( $numerek+1 ))

  # podzial pasma dostepu do sieci zewnetrznej (klasy identyfikowanej przez $htb_handle:2) na
  uzytkownikow
  tc class add dev $tc_down_dev parent $htb_handle:2 classid $htb_handle:$numerek htb \
   rate $curr_speed ceil $curr_speed_max

  # filtr ...
  for ip in `echo "$line" | cut -f1 -d'|'`; do
   tc_pref_inc 7
   tc filter add dev $tc_down_dev protocol ip   $TC_PREF parent $htb_handle:0 u32 \
    match ip  dst $ip flowid $htb_handle:$numerek
  done;
  for ip in `echo "$line" | cut -s -f2 -d'|'`; do
   tc_pref_inc 7
   tc filter add dev $tc_down_dev protocol ipv6 $TC_PREF parent $htb_handle:0 u32 \
    match ip6 dst $ip flowid $htb_handle:$numerek
  done;

  # sprawiedliwy dostęp do kabla wielu jednoczesnych połączeń
  tc qdisc add dev $tc_down_dev parent $htb_handle:$numerek sfq perturb 10
 done


# klasa ruchu niepriorytetowego - to czego gdzieindziej niezakwalifikujemy
 # (identyfikowana $htb_handle:7)
 # jest ona wskazana w parametrze "default" w "tc qdisc add ... htb"
 tc class add dev $tc_down_dev parent $htb_handle:2 classid $htb_handle:7 htb \
  rate 10kbit ceil 50kbit prio 10
 tc qdisc add dev $tc_down_dev parent $htb_handle:7 sfq perturb 10
}

stop_tc_down() {
 # kasujemy poprzednie ustawienia kolejek, filtrów, ...
 eval tc qdisc del root dev ${name[$ext_conf]} $QUITE
 eval tc qdisc del root dev imq1 $QUITE
}


########################
##     INIT.D SCRIPT     ##
########################

start_all() {
 # akceptacja pingów
 echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

```
echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_all

# tunele IPv6 i wstęp do routingu
 if [ "$extv6_conf" != ""  -a  "${name[$extv6_conf]}" != "" ]; then
  start_ipv6
fi
 start_routing

# ustawienia tymczasowe
 router_tmp

# firewall, NAT i kolejkowanie
 if [ "$ext_conf" != ""  -a  "${name[$ext_conf]}" != "" ]; then
  start_iptables
fi
 if [ "$extv6_conf" != ""  -a  "${name[$extv6_conf]}" != "" ]; then
  start_ip6tables
fi
 if $use_tc_down; then
  tc_pref_init
  start_tc_down
  tc_filter_tmp_down
fi
 if $use_tc_up; then
  tc_pref_init
  start_tc_up
  tc_filter_tmp_up
fi
}

stop_all() {
 stop_tc_up
stop_tc_down
stop_iptables
stop_ip6tables
stop_routing
stop_ipv6
}

# parametry skryptu

# DEBUG or NOT DEBUG ?
# aby to działało komendy w których wykorzystujemy to przekierowanie (zmienną $QUITE) ⤸
muszą być poprzedzone eval
QUITE=" 2> /dev/null"
if [ "$2" = "debug" -o "$2" = "moredebug" ]; then
 exec 2>&1
 if [ "$2" = "moredebug" ]; then
  QUITE=""
fi
 set -x
fi

# START, STOP or RESTART
case "$1" in
 start)
  start_all
  ;;
 stop)
  stop_all
```

```
  ;;
 restart)
  stop_all
  start_all
  ;;
 *)
  echo "Usage: $0 start|stop|restart [debug|moredebug]" >&2
  echo "e.g. $0 restart debug | grep -B 3 'RTNETLINK'"  >&2
  exit 3
  ;;
esac
```

ROUTING_FUN.SH

```
# generowanie priorytetów dla tc filter
tc_pref_init() {
 for (( i=0 ; $i<=9 ; i++ )) ; do
  PREFERENCE[$i]=0;
 done
}
tc_pref_inc() {
 if eval [ "$1" -le 9 -a "$1" -ge 0 ] $QUITE; then
  let PREFERENCE[$1]++
  TC_PREF=`printf "preference %d%03d" $1 ${PREFERENCE[$1]}`
else
  echo "ERROR: make_tc_prefernce: argument ($1) out of range (\$1 >= 0 AND \$1 <= 9)" >&2;
 fi
}

# konwersja maski (zapis skrócony) na zapis kropkowo dziesiętny
short2long_mask() {
 perl -e '
  use Socket;
  my $short_mask = $ARGV[0];
  if ($short_mask < 0 || $short_mask > 32) {
   exit;
  }
  if ($short_mask != 0) {
   $short_mask = 0x0ffffffff << (32-$short_mask);
  }
  print inet_ntoa(pack("N*", $short_mask));
 ' $1
}

# konwersja adresu IP (zapis kropkowo dziesiętny)
# i maski (skrócony, automatycznie rozwijany na kropkowo dziesiętny)
# na adres sieci (zapis kropkowo dziesiętny)
ip_and_mask2network() {
 perl -e '
  use Socket;

  # współcześnie powinno chodzić bez zabawy z tym pack / unpack
  my $ip = unpack("l*", pack("l*", unpack("N*", inet_aton($ARGV[0]))));
  my $mask = unpack("l*", pack("l*", unpack("N*", inet_aton($ARGV[1]))));

  print inet_ntoa(pack("N*", $ip & $mask));
 ' $1 `short2long_mask $2`
}
```

```
# konwersja adresu IPv6 i maski (skrócony) na adres sieci
ipv6_and_mask2network() {
 perl -e '
  use NetAddr::IP::Util qw(:inet :math);

  my $ip = ipv6_aton($ARGV[0]);
  my $mask = shiftleft(ipv6_aton("ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff"),128-$ARGV[1]);

  print ipv6_n2x($ip & $mask);
 ' $1 $2
}

# generacja /etc/network/interfaces
make_interfaces() {
 OUT="$1"

# utworzenie pliku z konfiguracją loopback
 echo -e 'auto lo\niface lo inet loopback' >$OUT

for X in `seq 0 $(( ${#mode[@]} - 1 ))`; do
  if [ "${mode[$X]}" = "int" ]; then
   echo "" >>$OUT
   echo "auto ${name[$X]}" >>$OUT
   echo "iface ${name[$X]} inet static" >>$OUT

   echo "  address ${base_ipv4[$X]}" >>$OUT
   mask=`short2long_mask ${base_maskv4[$X]}`
   echo "  netmask $mask" >>$OUT

   if [ "${gatewayv4[$X]}" != "" ]; then
    echo "  gateway ${gatewayv4[$X]}" >>$OUT
   fi

   count1=0;
   for ip in ${other_ipv4[$X]}; do
    count1=$(( $count1+1 ))
    count2=0
    for mask in ${other_maskv4[$X]}; do
     count2=$(( $count2+1 ))
     if [ $count2 = $count1 ]; then
      break
     fi
    done
    echo "  post-up ip address add broadcast + local $ip/$mask dev ${name[$X]}" >>$OUT
    echo "  pre-down ip address del $ip/$mask dev ${name[$X]}" >>$OUT
   done

   if [ "${base_ipv6[$X]}" != "" ]; then
    echo "iface ${name[$X]} inet6 static" >>$OUT
    echo "  address ${base_ipv6[$X]}" >>$OUT
    echo "  netmask ${base_maskv6[$X]}" >>$OUT
    if [ "${gatewayv6[$X]}" != "" ]; then
     echo "  gateway ${gatewayv6[$X]}" >>$OUT
    fi
    echo "   # konfiguracja ręczna (wyłączamy automatyczną)" >>$OUT
    echo "   pre-up  echo 0 > /proc/sys/net/ipv6/conf/${name[$X]}/autoconf" >>$OUT
    echo "   # zmijszamy MTU 9ważne przy połączeniach tunelowanych)" >>$OUT
    echo "   pre-up  echo 1280 > /proc/sys/net/ipv6/conf/eth0/mtu" >>$OUT
```

```
      count1=0;
      for ip in ${other_ipv6[$X]}; do
       count1=$(( $count1+1 ))
       count2=0
       for mask in ${other_maskv6[$X]}; do
        count2=$(( $count2+1 ))
        if [ $count2 = $count1 ]; then
         break
        fi
       done
       echo "  post-up ip address add local $ip/$mask dev ${name[$X]}" >>$OUT
       echo "  pre-down ip address del $ip/$mask dev ${name[$X]}" >>$OUT
      done
     fi
    fi
  done
}
```

ROUTING.CONF

```
# example routing config file

####################
##   INTERFEJSY   ##
####################

# indeksy tablicy konfiguracyjnej interfejsow dla:
#  interfejsu wewnętrznego
int_conf=0
# interfejsu zewnętrznego
ext_conf=1
# interfejsu zewnętrznego dla ipv6
extv6_conf=2


# typ konfiguracji o numerze X int = interfejs, sit = tunel typu sit
mode[0]="int"
# nazwa interfejsu tunelu o numerze
name[0]="eth0"
# prędkość urządzenia dla traffic control
interface_speed[0]="99mbit"

# podstawowy numer IPv4
base_ipv4[0]="192.168.1.1"
# długość prefixu (maska w postaci ilości bitów o wartości 1) podstawowego numeru IPv4
base_maskv4[0]="24"
# bramka domyślna IPv4
#gatewayv4[X]="IP.IP.IP.IP"
# dodatkowe adresy IPv4 rozdzielane spacjami
other_ipv4[0]="192.168.1.254 192.168.10.1"
# długości prefixów (maski w postaci ilości bitów o wartości 1) dodatkowych adresów ⤸
IPv4 rozdzielane spacjami (w kolejności takiej jak adresy)
other_maskv4[0]="24 24"

# podstawowy numer IPv6
base_ipv6[0]="2001:db8::1"
# długość prefixu (maskaw postaci ilości bitów o wartości 1) podstawowego numeru IPv6
base_maskv6[0]="48"
```

```
# bramka domyślna IPv6
# gatewayv6[X]="IP:IP::IP"
# dodatkowe adresy IPv6 rozdzielane spacjami
other_ipv6[0]="2001:db8::5879:12df"
# długości prefixów (w postaci ilości bitów o wartości 1) dodatkowych adresów IPv6 ⤸
rozdzielane spacjami (w kolejności takiej jak adresy)
other_maskv6[0]="48"

# adres IPv4 drugiego końca tunelu (tylko dla sit)
#remote[X]="IP.IP.IP.IP"
# adres IPv4 tego końca tunelu (tylko dla sit)
#local[X]="IP.IP.IP.IP"


mode[1]="int"
name[1]="eth1"
base_ipv4[1]="10.10.0.1"
base_maskv4[1]="24"
# bramka domyślna IPv4
gatewayv4[1]="10.10.0.2"


mode[2]="sit"
name[2]="tun1"
base_ipv6[2]="2001:db8:1::5"
base_maskv6[2]="126"
remote[2]="10.10.2.1"
local[2]="10.10.0.1"



##################
##   IPTABLES   ##
##################

# rozdzielana spacjami lista blokowanych portów
block_ports="98 143 10000 20000"

# typ NATu - SNAT (gdy na interfejsie z stałym IP) lub MASQUERADE (gdy na interfejsie z ⤸
zmiennym IP) lub NO (aby nie korzystać z natu)
nat_mode="SNAT"
# sieć NATowana
nat_network="192.168.1.0/24"

# adresy IP (ewentualnie sieci w postaci adres.sieci/maska_krotka)
#   uprawnione do wyjścia na świat, można wczytać z pliku np. forwarding_ip=`<plik.conf`
#   każda linia jest wrzucana do jednej kolejki HTB, składa się z 4 pól rozdzielanych | ->
#   lista adresów IPv4 (rozdz. spacjami), lista adresów IPv6 (rozdz. spacjami),
#   ograniczenia HTB - prędkość gwarantowana, ograniczenia HTB - prędkość maksymalna
forwarding_ip=`cat /etc/forwarding_ip.conf`
forwarding_ip="$forwarding_ip
192.168.1.0/26|2001:db8::341 2001:db8::342|100kbit|90kbit
192.168.1.244||100kbit
192.168.1.245"

#  tymczasowe i spacjalne ustawienia routingu (np. iptables)
#   mogą to być wszczególności przekierowania portów dla połączeń p2p:
#    iptables -A PREROUTING -t nat -d $IP_ZEWNETRZNE -p tcp --dport $PORT_ZEWNETRZNY \
#    -j DNAT --to $IP_WEWNETRZNE:$PORT_WEWNWTRZNY
#  a także inne ustawienia, jak tymczasowe blokady itp
```

```
router_tmp(){
# ograniczenie ilości pingów do 16 na minute ...
# sprzątamy
iptables -F PINGI 2> /dev/null
iptables -X PINGI 2> /dev/null
# wszystkie tworzymy osobny łańcuch na pingi
iptables -N PINGI
# kierujemy do niego wszystkie pingi
iptables -A INPUT -p icmp --icmp-type echo-request -j PINGI
# puszczamy 16 na minute
iptables -A PINGI -m limit --limit 16/minute -j ACCEPT
# reszte DROPujemy
iptables -A PINGI -j DROP
# powyższe ilustruje też tworzenie własnego łańcucha i kierowanie ruchu do niego
# zamiast powtarzania reguły filtrującej (-p icmp --icmp-type echo-request)
# w każdej z reguł mających normalny TARGET (w tym wypadku regół działających na ⤸
łańcuchu PINGI)

# dostęp do SMTP z LANu na niestandardowym porcie (zab. pred robakami wysyłającymi spam)
iptables -A PREROUTING -t mangle -s 192.168.0.0/16 -d 0.0.0.0/0 -p tcp --dport 325 -j MARK --⤸
set-mark 0x01
iptables -A PREROUTING -t mangle -s 192.168.0.0/16 -d 0.0.0.0/0 -p tcp --dport 25  -j MARK --⤸
set-mark 0x00
iptables -A PREROUTING -t nat    -s 192.168.0.0/16 -d 0.0.0.0/0 -p tcp --dport 325 -j DNAT --⤸
to :25
iptables -A FORWARD -s 192.168.0.0/16 -d 0.0.0.0/0 -p tcp --dport 25 -m mark --mark 0x01 -j ⤸
ACCEPT
iptables -A FORWARD -s 192.168.0.0/16 -d 0.0.0.0/0 -p tcp --dport 25 -j DROP
# wykorzystane w powyższym markowanie można także sprawdzać/wykorzystać w:
# - filtrach tc (patrz poniżej)
# - w wyborze tablicy routingu (ip rule ... fwmark)
# - itd

# przyjmujemy tylko ssh
# iptables -A INPUT -p tcp --dport 22 -j ACCEPT
# iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# iptables -A INPUT -p tcp -j DROP

# forward pomiędzy dwiema klasami adresowymi LANu
# w zasadzie należy konfigurować na każdym hoście na urządzeniu routing na dwie klasy
# chyba że są to różne vlany lub różne fizyczne lany
iptables -A FORWARD -i ${name[$int_conf]} -s $nat_network -d 192.168.10.1/24 -j ACCEPT
iptables -A FORWARD -i ${name[$int_conf]} -s 192.168.10.1/24 -d $nat_network -j ACCEPT

# przekierowanie portu do LANU
iptables -A PREROUTING -t nat -d ${base_ipv4[$ext_conf]} -p tcp --dport 8024 -j DNAT --to ⤸
192.168.1.80:8010

# MTU dla tuneli IPv6
# ifconfig eth1 mtu 1280
ifconfig tun1 mtu 1280
}



##########################
##    TrafficControl    ##
##########################

# czy korzystać z IMQ (wymaga to kernela skompilowanego z obsługą IMQ w trybie AB)
```

```
use_imq=true

# czy korzystać z kolejki priorytetowej umieszczoneuj "powyżej" htb
use_prio_queue=true

# host z którego filtryujemy po ToS dla potzeb ruchu priorytetowego
voip_host_ipv4="10.10.0.1"
voip_host_ipv6="2001:db8::341 2001:db8::456"


# czy włączone tc (htb) dla uploadu
use_tc_up=true
# pełna prędkość urządzenia zewnętrznego dla traffic control (do ISP)
full_speed_up="3010898"
# prędkośc gwarantowana kolejki priorytetowej
pri_speed_up="160kbit"
# prędkość urządzenia zewnętrznego po odjęciu rezerwy (do podziału na uzytkowników)
speed_up="2700000"
# prędkość gwarantowana uploadu dla sieci serwerowej
serw_speed_up="1000kbit"
# prędkość gwarantowana uploadu dla sieci LAN
lan_speed_up="500kbit"

#  dodatkowe ustawienia filtru pakierow dla traffic control dla uploadu
#   np. filtrowanie po portach (u32 match ip [dport|sport])
#    lub polu TOS (u32 match ip tos ...)
#    lub na podstawie znaczników fwmark uswawianych przez -j MARK --set-mark (handle)
#  zamiast "preference PREF" należy korzystać z $TC_PREF ustawianej wywołaniem ⤸
"tc_pref_inc NUM"
#  gdzie NUM odpowiada liczbie całkowitej >=0 i <=9, standarowe wartości NUM:
#   1 - wewnetrzne
#   3 - priorytetowe
#   5 - serwery
#   7 - reszta sieci
tc_filter_tmp_up() {
# uprzywilejowanie pingów z wybranego hosta
tc_pref_inc 3
tc filter add dev $tc_up_dev protocol ip $TC_PREF parent 2:0 u32 match ip protocol 1 0xff ⤸
match ip src $voip_host_ipv4 flowid 2:4

# operowanie na ruchu oznaczonym poprzez "-j MARK --set-mark 0x03" w iptables
# tc_pref_inc 3; tc filter add dev $tc_up_dev protocol all $TC_PREF parent 2:0 handle 3 fw ⤸
flowid 2:4
}


# czy włączone tc (htb) dla downloadu
use_tc_down=true
# różnica prędkości urządzenia wewnętrznego i łącza (prędkość kolejki ruchu ⤸
wewnętrznego)
int_speed_down="95mbit"
# pełna prędkość urządzenia zewnętrznego dla traffic control (od ISP)
full_speed_down="3010898" # 2.88 * 1024 * 1024 = 3019898
# prędkośc gwarantowana kolejki priorytetowej
pri_speed_down="160kbit"
# prędkość urządzenia zewnętrznego po odjęciu rezerwy (do podziału na uzytkowników)
speed_down="2700000"
# minimalne (gwarantowane) prędkości dla pojedynczych użytkowników (grup IP zaliczanych ⤸
do kolejki pojedyńczego użytkownika)
user_speed_down="100kbit"
```

```
#  dodatkowe ustawienia filtru pakierow dla traffic control dla downloadu
#   np. filtrowanie po portach (u32 match ip [dport|sport])
#    lub polu TOS (u32 match ip tos ...)
#    lub na podstawie znaczników fwmark uswawianych przez -j MARK --set-mark (handle)
#  zamiast "preference PREF" należy korzystać z $TC_PREF ustawianej wywołaniem ⤸
"tc_pref_inc NUM"
#  gdzie NUM odpowiada liczbie całkowitej >=0 i <=9, standarowe wartości NUM:
#   1 - wewnetrzne
#   3 - priorytetowe
#   5 - serwery
#   7 - reszta sieci
tc_filter_tmp_down() {
# filtracja w oparciu o protokół - ICMP (numery w /etc/protocols)
tc_pref_inc 3
tc filter add dev $tc_down_dev protocol ip $TC_PREF parent 1:0 u32 match ip protocol 1 0xff ⤸
flowid 1:4
# filtracja w oparciu o IP przeznaczenia
tc_pref_inc 3
tc filter add dev $tc_down_dev protocol ip $TC_PREF parent 1:0 u32 match ip dst ⤸
$voip_host_ipv4 flowid 1:4
# w odróżnieniu od regóły zaprezentowanej w tc_filter_tmp_up() tutaj uprzywilejowany ⤸
jest cały ruch ICMP i cały ruch do hosta $voip_host

# filtracja w oparciu o ToS
tc_pref_inc 3
tc filter add dev $tc_down_dev protocol ip $TC_PREF parent 1:0 u32 match ip tos 0x10 0xff ⤸
flowid 1:4

# filtracja w oparciu o port - SSH (uwaga po ssh można przesyłać w zasadzie wszystko, ⤸
także duże pliki więc jego uprzywilejowywanie jest kontrowersyjne)
tc_pref_inc 3
tc filter add dev $tc_down_dev protocol ip $TC_PREF parent 1:0 u32 match ip dport 22 0xffff ⤸
flowid 1:4
}
```

TRAFFIC_STATS.SH

```
#!/bin/bash

#
# skrypt generujący statystyki ruchu
# (zaklada sie ze udostepnianie lacza odbywa sie poprzez skrypt router.sh mojego autorstwa)
#

okres_probkowania=30
identyfikator="predkosci"

if [ ! -f /tmp/${identyfikator}_out.txt ]; then
 :> /tmp/${identyfikator}_out.txt
fi

while true; do
 # dane o przeslanych bajtach na IP z iptables -L
 # uwaga: ruch bedacy FORWARD nie jest zaliczany do INPUT/OUTPUT
 #   - przy wylaczonym INPUT/OUTPUT FORWARD dziala ...
 iptables -L FORWARD -n -v -x > /tmp/${identyfikator}_in.txt
iptables -L INPUT -n -v -x >> /tmp/${identyfikator}_in.txt
```

```
iptables -L OUTPUT -n -v -x >> /tmp/${identyfikator}_in.txt
# AWK ...
 env TZ=UTC awk -v plik_out="/tmp/${identyfikator}_out.txt" '
  # dane z iptables
  $3=="ACCEPT" && $8!="0.0.0.0/0" && $9=="0.0.0.0/0" { wysylane[$8]=$2 }
  $3=="ACCEPT" && $8=="0.0.0.0/0" && $9!="0.0.0.0/0" { odebrane[$9]=$2 }
  # dane z poprzedniego wywolania skryptu
  $3!="ACCEPT" { wysylane_OLD[$1]=$2; odebrane_OLD[$1]=$3 }
  # zapisywanie wynikow
  END {
   #zapis nowych danych (do plik_out)
   czas=systime()
   for(INDEX in wysylane) {
    if (! INDEX in odebrane)
     odebrane[INDEX]=0
   }
   for(INDEX in odebrane) {
    if (! INDEX in wysylane)
     wysylane[INDEX]=0
    printf("%s\t%s\t%s\n", INDEX, wysylane[INDEX], odebrane[INDEX]) > plik_out
   }
   printf("TIME\t%s\n", czas) > plik_out

   #zapis statystyki - roznica (na stdout)
   czas=czas-wysylane_OLD["TIME"]
   #printf "IP\t\twysylane\todbierane (z Internetu)\n"
   for(INDEX in wysylane)
    printf("%s\t%9.2f\t%9.2f\n", INDEX, (wysylane[INDEX]-wysylane_OLD[INDEX])/czas, \
     (odebrane[INDEX]-odebrane_OLD[INDEX])/czas)

   printf("OKRES PROBKOWANIA:\t%d s\n", czas)
   printf("DATA:\t%s\n", strftime("%Y-%m-%d %H:%M:%S %Z (%s)"))
  }
 ' /tmp/${identyfikator}_in.txt /tmp/${identyfikator}_out.txt > /var/log/${identyfikator}.log
# spimy ...
 sleep $okres_probkowania
done
```

TRAFFIC_STATS.INIT.SH

```
#! /bin/sh

PATH=/sbin:/usr/sbin:/bin:/usr/bin


case "$1" in
    start)
        /usr/local/bin/traffic_stats.sh &
        ;;
    stop)
        killall traffic_stats.sh
        ;;
    *)
        echo "Usage: $0 start|stop" >&2
        exit 3
        ;;
esac
```

TRAFFIC_STATS_SHOW.SH

```
#!/bin/bash

#
# skrypt generujący ładniejsze statystki (dla WWW) na podstawie /var/log/predkosci.log
#    generowanego przez skrypt statystyki_ruchu.sh
#

echo "Content-type: text/plain"
echo ""

echo "WLACZONE ADRESY IP:"
echo "==================="
sort -n -t . -k 1,1 -k 2,2 -k 3,3 -k 4,4 /proc/net/arp | grep -v eth1

echo ""
echo ""
echo ""

echo "PREDKOSCI REALNE (w bajtach/s, srednie z 30 sek):"
echo "================================================="
echo -e "IP\t\twysylane\todbierane (z Internetu)"
lista_ip=`ip addr sh | awk '
 $1=="inet" {
 split($2, tmp, "[./]");
 printf("(%s",tmp[1])
 for (i=2; i<=tmp[5]/8; i++)
  printf(".%s", tmp[i])
 printf(")")
 }' | sed -e 's#)(#)|(#g'`
grep -v '^[A-Z]' /var/log/predkosci.log | egrep "$lista_ip" |  sort -n -t . -k 1,1 -k 2,2 -k ⤸
3,3 -k 4,4
echo ""
grep '^[A-Z]' /var/log/predkosci.log

exit
```

Routing

Istotną funkcją w różnych systemach opartych o protokół IP jest routing. Często jest on realizowany przez wyspecjalizowane urządzania (routery sprzętowe), ale równie często jego realizację przeprowadza się software'owo. Zaczęcam do zapoznania się z zestawem skryptów do konfiguracji routingu i podziału pamsa mojego autorstwa.

W tym miejscu warto wspomnieć o dość interesującej alternatywie w stosunku co do IMQ jaką jest obecne standardowo w jądrze IFB. Inną alternatywą jest lokalne tunelowanie ruchu (tunel na interfejsie lo). Więcej na jego temat takich rozwiązań znaleźć można w: jak radzić sobie bez IMQ?[1750] i IMQ bez IMQ HOWTO[1751].

Warto zaznaczyć iż do przekierowywania usług/portów mogą być realizowane z wykorzystaniem xinetd (zobacz konfigurację) a nie DNAT - co nie rodzi problemów z IPv6 oraz dwiema sieciami IP na jednym interfejsie Kolejnym popularnym zastosowaniem NATu jest chęć skierowania ruchu z jakiś adresów do danego hosta (wyświetlającego np. komunikat odnośnie przerwy w dostępie do Internetu). W przypadku IPv6, gdzie (na linuxie) nie mamy do dyspozycji NATu, możemy to zrobić modyfikując działanie DNS (odpalając sfałszowany DNS) tak aby na każde zapytanie odpowiadał adresem naszego hosta z komunikatem. Sam serwer DNS może identyfikować czy hosta należy przekierować czy nie lub możemy to oprzeć np. na różnych adresach DNS wysyłanych przez DHCP do wybranych grup hostów.

Wspomniana technologia NAT (o której troszkę więcej w artykule o sieciach IP) korzysta z tablicy w której przechowuje śledzone połączenia (pamiętanie NATowanych połączeń jest konieczne aby wykonywać NAT). Aktualną ilość wpisów w tej tablicy można sprawdzić poprzez `wc -l < /proc/net/ip_conntrack`. Gdy tablica ta się zapełni nie jest możliwe nawiązywanie nowych połączeń, gdy występuje taki pro-

blem możemy zmienić jej rozmiar lub zmniejszyć czas przechowywania połączeń poprzez ustawienie odpowiednich wartości odpowiednio w /proc/sys/net/ipv4/netfilter/ip_conntrack_max /proc/sys/net/ipv4/netfilter/ip_conntrack_tcp_timeout_established. Interesujący wydaje się także patch[1752] umożliwiający wybór typu NAT przy konfiguracji jądra.

Częstym aspektem przy konfiguracji bramek (jednak tutaj praktycznie pominiętym) jest autoryzacja klientów uprawnionych do dostępu do sieci. Najpopularniejszym rozwiązaniem tego typu jest stosowanie PPPoE[1753], spotkać się można także z wykorzystaniem autoryzacji proxy[1754], stosowania VPN w tym celu. W systemach BSD możliwe jest także dopuszczanie użytkowników którzy nawiążą z serwerem połączenie SSH - poprzez mechanizm Authpf.

W IPv6 niezaniedbywalnym elementem routingu jest rozgłaszanie inforamacji routingowych poprzez mechanizm typu radv - może do tego posłużyć serwer "radvd", a jego przykładowa konfiguracja (/etc/radvd.conf) może wyglądać np. następująco:

```
interface eth0
{
        # konfiguracja adresów stateful (TAK)
        AdvManagedFlag on;
        # konfiguracja innych parametrów stateful (NIE)
        AdvOtherConfigFlag off;
        # wysylamy MTU
        AdvLinkMTU 1280;
        # wysylamy okresowo
        AdvSendAdvert on;
        #AdvSourceLLAddress off
        prefix 2001:db8:1000:3b::/64
        {
                # pytanie czy i jak da się skutecznie rozgłosić sieć z prefiksem np. /48
↪   ?
                # tak aby jej wybrany fragment stanowił sieć autokonfigurowaną ale maska
↪   dla global-scope była /38 ...
                AdvOnLink on;
                AdvAutonomous on;
        };
        #route 2001:db8:1000::1/48 {
        #        AdvRoutePreference medium;
        #};
        route 0::/0 {
                AdvRoutePreference medium;
        };
};
```

Kożystanie z protokołów routingowych takich jak SPF, BGP, czy też korzystanie z routingu multicastowego wymaga stosowania deamonów zajmujących się obsługą tych protokołów i ustawiających w oparciu o nie odpowiednich jądrowych tablic routingu. Dla różnych systemów POSIXowych Quagga[1755] (Quagga[1756] jest nieoficjalnym kontynuatorem projektu GNU Zebra[1757]), xorp[1758], bird[1759], dla OpenBSD (i z drobnymi łatami dla innych *BSD) jest to na przykład OpenBGPD[1760]. Dla protokołów multicastowych są to np. mrd6[1761], Multi-threaded Routing Toolkit[1762], mrouted[1763] i pimd[1764].

Zobacz w Sieci: Nftables: a new packet filtering engine[1765], Jak podróżuje pakiet przez router linuxowy[1766], HTB, filtry u32 i inne[1767], Linux-Mrouted-MiniHOWTO[1768].

REDIRECT-XINETD

```
# bind = ::0 jest sztuczką powodującą że usługa jest dostępna przez IPv6 oraz przez ⤸
IPv6 ...
# niestety dostęp po IPv4 śmieci wtedy w logach: "Setting IPV6_V6ONLY option failed"
# jeżeli nie podano tej opcji usługa dostępna jest tylko po IPv4
# (chyba że podano flagę IPv6, ale nie jest możliwe podanie flag IPv4 i IPv6 jednocześnie ⤸
...)
# wersja pakietu xinetd: 2.3.14-1
service www
{
```

```
    bind           = ::0
    socket_type    = stream
    wait           = no
    user           = root
    redirect       = moj.serwer.www 80
}

# wymaga wpisania do /etc/services :
# xmpp-sclient  5223/tcp jabber-client # Jabber Client Connection
# xmpp-sclient  5223/udp jabber-client
service xmpp-sclient
{
    bind           = ::0
    socket_type    = stream
    wait           = no
    user           = root
    redirect       = moj.serwer.xmpp 5223
}


service xmpp-client
{
    bind           = ::0
    socket_type    = stream
    wait           = no
    user           = root
    redirect       = moj.serwer.xmpp 5222
}


service xmpp-server
{
    bind           = ::0
    socket_type    = stream
    wait           = no
    user           = root
    redirect       = moj.serwer.xmpp 5269
}
```

DHCP

Zamieszczam przykładowe konfigi dla DHCP zarówno w wersji dla IPv4 - `/etc/dhcp3/dhcpd.conf` (konfig przystosowany jest do współpracy z systemem konfiguracji przez użytkownika):

```
default-lease-time 600;
max-lease-time 7200;
authoritative;
option classless-static-route code 121 = array of { integer 8, ip-address, ip-address };
option classless-static-route-win code 249 = array of { integer 8, ip-address, ip-address
 ↪  };
subnet 192.168.0.0 netmask 255.255.0.0 {
        range 192.168.28.1 192.168.28.100;
        option subnet-mask 255.255.0.0;
        option broadcast-address 192.168.255.255;
        option routers 192.168.0.1;
        option domain-name-servers 192.168.0.1;
        # przydatne gdy w sieci lokalnej mamy adresy prywatne i publiczne
        option classless-static-route MASKA_KROTKA IP_SIECI_PUB 0.0.0.0;
```

```
        option classless-static-route-win 32 ADRES_HOSTA_PUB ADRES_HOSTA_PRIV, 32
 ↪  ADRES_HOSTA_PUB ADRES_HOSTA_PRIV;
}
# konfiguracja centralna ...
host h1{
        hardware ethernet XX:XX:XX:XX:XX:XX;
        fixed-address YYY.YYY.YYY.YYY;
}
# konfiguracja generowana z plików użytkowników
include "/var/user_config/dhcpd.conf";
```

A także dla IPv6 - `/etc/wide-dhcpv6/dhcp6s.conf`:

```
option domain-name-servers 2001:db8:1000:3a::1000;
option domain-name "n17.waw.pl"
# jako klient najprzyjemniejszy wydaje się dibbler-client
# ale serwer wide-dhcpv6 jest bardziej elegancki w konfiguracji (zwłaszcza stały adres
 ↪  dla wskazanego hosta)
host robert {
        duid unik:alnyident:yfika:tor:dhcp;
        # UWGA: nie jest to MAC adres !!!
        # jest on stały dla systemu a nie katrty sieciowej (czyli w większości wypadków
 ↪  mniej stały niż MAC)
        address 2001:db8:1000:3a::2 infinity;
        # z jakiś przyczyn adres zostanie przyjęty jako pochodzący z sieci /64 i na
 ↪  ddatek nie ma niestyety:
        # * jawnej mozliwości rozgłaszania długości prefiksu :-(
        # * rozgłaszania konfiguracji routingu
};
interface eth0 {
        address-pool pool1 3600;
};
pool pool1 {
        range 2001:db8:1000::1 to 2001:db8:1000::100 ;
};
```

NAT-PT czyli brama pomiędzy IPv6 a IPv4

Przynajmniej w okresie przejściowym wdrażania IPv6 konieczne może się okazać zapewnienie sieci IPv6-only dostępu do zasobów sieci IPv4. W tym celu stosuje się rozwiązania typu NAT-PT[1769], wykonujące translację adresów pomiędzy IPv6 i IPv4. W odróżnieniu od zwykłych NATów muszą one także podmieniać zapytania i odpowiedzi DNS tak aby mapować adres IPv4 hosta docelowego w przestrzeni adresowej IPv6 i zwracać ten zamapowany adres jako odpowiedź na rządanie AAAA.

Przykładem realizacji takiego mechanizmu jest daemon (mechanizm ten - chyba nawet w BSD oferującym w odróżnieniu od Linuxa NAT dla IPv6 - realizowany jest poza jądrem systemu) naptd[1770]. Instalacja i konfiguracja jest szczegółowo opisana w załączonej dokumentacji. Należy jednak zwrócić uwagę blokadę przekazywania prefixu używanego przez NAT-PT na routerach IPv6 - `ip6tables -I FORWARD 1 -d 2000:ffff::/96 -j DROP` (na wszystkich routerach dających wyjście na świat po ipv6) oraz odrzucanie pakietów innych niż ESTABLISHED oraz NEW na uruchomione usługi (ptrzedewszystkim ssh) na routerze robiącym NAT-PT. Na hostach klienckich należy z koleji ustawić trasę routingu dla sieci wykorzystywanej przez NAT-pT - `ip -6 route add 2000:ffff::/96 via ipv6:hosta:robiacego:nat::pt dev eth0` oraz ustawić DNS będący adresem DNSu IPv6 zamapowanym na adres IPv6 urzywany przez NAT-PT (2000:ffff::ip.ip.dns.v4). Zamieszczam też łatkę[1771] na wersję 0.4 umożliwiającą skompilowanie jej pod Debianem Lenny.

Inne ustawienia i uwagi

Mogłoby się wydawać iż oprócz logowania ruchu przechodzącego (omawianego w części dotyczącej routingu) warto rozważyć logowanie tablicy arp (odpowiedniości numerów sprzętowych i IP) - realizuje to np. następujący skrypcik - arp_log.sh, jednak po bliższym przyjżeniu się logom ruchu zauważyć w nich można oprócz adresów

IP MAC adresy kart, więc jest to zbędne. Na koniec proponuję dodanie do /etc/cron.daily/ skryptu wywołującego `ntpdate` (uwaga: aby skrypt był wykonywany przez crona konieczne może okazać się zrezygnowanie z niektórych znaków w jego nazwie np. kropki).

Przydatne może być też zmienienie portu na którym słucha SSH ze względu na boty próbujące się logować i zaśmiecające nam logi. Niekiedy warto także logować połączenia SSH w lastlogu, aby to uczynić należy do `/etc/pam.d/sshd` dodać `session    optional    pam_lastlog.so`.

W przypadku problemów z wstaniem mysql po przenoszeniu syustemu pomocne moze byc jego odinstalowanie i ponowne zainstalowanie oraz zaimportowanie sql (przy usuwaniu kasujemy takze pliki z bazai danych). W przypadku problemów z logowaniem do phpmyadmin należy usunąć ciasteczka.

Zachęcam także do zapoznania się z moimi artykułami o: konfiguracji sieci i konfiguracji systemów operacyjnych (w szczególności o konfiguracji logów systemowych).

Konfiguracja w ręce użytkowników

Jeżeli zależy nam na samodzielności użytkowników możemy spore możliwości konfiguracyjne przekazać w ich ręce. W ogólności konfiguracja użytkownika może być wprowadza w zależności od usługi na różne sposoby - w przypadku tradycyjnych usług jak poczta i www może to być dostęp shell'owy (ssh), w przypadku jabber'a (XMPP) jest to konfiguracja przez ten protokół (konsola XML/klient) w przypadku voip (SIP) możemy dokonywać konfiguracji poprzez telefon (system menu telefonicznego).

W świetle przedstawionych tutaj rozwiązań warto również rozważyć sposób stworzenia bazy domen z powiązaniem do UID/GID (tabelka domena UID, GID jeden z UID/GID == x, mająca znaczenie informacyjne, bo automatyczne generowanie konfigu z plików użytkownika odpada) a także pamiętać też należy że wywołania adduser z odpowiednimi opcjami umożliwiają określenie identyfikatorów numerycznych jakie przypisujemy użytkownikowi/grupie:

- `adduser --group --gid GROUP_NUM_IDGROUP_NAME`
- `adduser --gid GROUP_NUM_ID [--add_extra_groups ...] USER_NAME`
- `adduser --ingroup GROUP_NAME [--add_extra_groups ...] USER_NAME`

W przypadku omawianego w tym serwisie serwera ejabberd warto także zapewnić okresowy zrzut jego bazy danych (przechowującej m.in. ustawienia wprowadzane przez użytkowników do pliku tekstowego. Można tego dokonać okresowo wywołując: `ejabberdctl dump /tmp/ejabberd_`date +%Y%m%dT%H%M%S`.txt`. Można by także spróbować parsować tak otrzymany plik celem poumieszczania zrzutu konfiguracji w katalogach odpowiednich użytkowników.

Można by stworzyć system generujący aliasy XMPP (zobacz: JID aliases) i aliasy SIP w oparciu o aliasy pocztowe (virtusertable). Główną wątpliwość budzi fakt iż raczej dość często mogą być potrzebne aliasy tylko jednego z rodzajów a nie wszystkich na raz więc może jednak lepiej utrzymywać niezależne bazy aliasów dla SMTP, XMPP i SIP. Dlatego też jak na razie nie został on prze zemnie przygotowany, ale być może kiedyś to nastąpi i go tu opublikuję.

Oczywiście jak mówimy o konfiguracji to trzeba wspomnieć o programach takich jak `chsh` i `chfn`, umożliwiających użytkownikowi odpowiednio zmianę używanego przez niego shella oraz danych zapisanych w polu informacji dodatkowych w `/etc/passwd`. Za ograniczenie sheli na jakie może być dokonywana zmiana odpowiada plik `/etc/shells` a za ograniczenie pól opisu w bazie kont które może edytować użytkownik odpowiada wpis `CHFN_RESTRICT` w `/etc/login.defs`. O `passwd` nie będę pisał ;-).

DHCP

Inną usługą gdzie możemy chcieć aby użytkownik mógł samemu mógł wprowadzać dane konfiguracyjne jest DHCP - zamieszczam skrypt generujący w oparciu o listę mac-adresów w pliku użytkownika konfigurację dla serwera dhcp. Założenie działania skryptu jest takie iż identyfikator numeryczny grupy podstawowej użytkownika (dokładniej grupy będącej właścicielem pliku konfiguracyjnego) przekłada się w prosty sposób na system numeracji IP. Zobacz także prototypy skryptów umożliwiających dodawanie MAC adresów przez WWW - add_mac.php add_mac.sh.

System wymaga aby dodać do `/etc/sudoers` następujące linie:

```
www-data ALL =NOPASSWD: /usr/local/bin/add_mac.sh
ALL ALL =NOPASSWD: /usr/local/bin/dhcp_user_conf.sh
```

W przypadku gdy zainstalowany jest usermin możliwe jest stworzenie modułu do edycji listy mac-adresów.

Prezentowany skrypt przygotowany został do utworzenia takiego modułu (bazującego na module ”plan”) dla usermin’a w wrsji 1.340.

Uwaga: prezentowane powyżej programy mają status beta - przechodzą one dopiero testy i mogą pojawić się jakieś błędy.

DHCP_USER_CONF.SH

```
#!/bin/bash

# skrypt generujący sumaryczna konfiguracja DHCP w oparciu o MAC'adresy podane przez ⤸
użytkownika
# plik wejściowy składa się z wpisów postaci: ADRES:MAC koncowkaIP
# linie rozpoczynające się od # są ignorowane

PATH="/usr/local/bin:/usr/bin:/bin"

USER_DHCP='/var/user_config/dhcpd.conf'
MIN_GID=1000
BASE_IP="192.168"
DHCP_IP_START=100

echo "# plik generowany automatycznie - proszę nie edytować" > $USER_DHCP
for f in /home/*/.dhcp_mac; do
 # uzyskujemy identyfikator grupy do IP ktorej przypisane mają być podane MAC' adresy
 HOME_DIR=`dirname "$f"`
MAC_GID=$(( $(id -g `ls -ld "$HOME_DIR" | cut -f3 -d' '`) - $MIN_GID ))

# czytamy plik z MAC adresami i generujemy konfig DHCP
 grep -v '^#' < "$f" | while read MAC ITER; do
  if [ "${ITER}" != "" ]; then
   if [ ${ITER} -lt 255 ]; then
    echo "" >> $USER_DHCP
    echo "host l${MAC_GID}h${ITER}" >> $USER_DHCP
    echo " { hardware ethernet $MAC; fixed-address ${BASE_IP}.${MAC_GID}.${ITER}; }" >> ⤸
    $USER_DHCP
   fi
  fi
 done
done

# w głównym pliku należy użyć
#    include sciezka do pliku generowanego przez skrypt

/etc/init.d/dhcp3-server restart
```

ADD_MAC.PHP

```
<?php
/** Skrypt umożliwia logowanie się do systemu phpmyadmin uzywając hasła i loginu konta ⤸
shelowego
 *    na którym umieszczony jest stosowny plik .my.cnf zawierający nazwę uzytkownika i ⤸
 hasło MySQL
 **/

header('Content-Type: text/html; charset=utf-8', 1);
?><html>
```

```
<head>
  <title>Rejestrowanie nowego komputera w sieci n17.waw.pl</title>
 </head>
 <body>
<?php
$username = $_POST['username'];
 $password = $_POST['password'];

if ($username != "") {
  if (! pam_auth($username, $password, &$error)) {
   echo $error;
   exit;
  }

  if ($_GET['action'] == "reload") {
   echo "<p>Proszę czekać trwa przeładowywanie ustawień DHCP</p><pre>";
   system("sudo /usr/local/bin/dhcp_user_conf.sh");
   echo "</pre><p>Ustawienia DHCP przeładowane, proszę odnowić adres IP w kliencie</p>";
  } else {
   $mac = shell_exec ("/sbin/ip neigh sh | egrep '^" . $_SERVER['REMOTE_ADDR'] . " ' | awk '{
   printf(\"%s\", $5)}'");
   echo "<p>Logowanie OK.</p>\n";
   echo "<p>Rejestrowanie $mac dla użytkownika: $username</p>\n<pre>";
   system("sudo /usr/local/bin/add_mac.sh $mac $username");
   echo "<pre>";
  }
 } else {
  echo '
  <p>W poniższe pola należy wpisać swój login i hasło do konta shelowego</p>
  <center>
  <form method="post" action="?action='. $_GET['action'] .'" name="login" target="_top">
  <table cellpadding="3" cellspacing="0">
   <tr>
    <td align="right" bgcolor="#E5E5E5"><b>Użytkownik: </b></td>
    <td align="left" bgcolor="#E5E5E5">
     <input type="text" name="username" size="24" class="textfield" />
    </td>
   </tr>
   <tr>
    <td align="right" bgcolor="#E5E5E5"><b>Hasło: </b></td>
    <td align="left" bgcolor="#E5E5E5">
     <input type="password" name="password" size="24" class="textfield" />
    </td>
   </tr>
   <tr>
    <td colspan="2" align="center">
     <input type="submit" value="Login" id="buttonYes" />
    </td>
   </tr>
  </table></form></center>';
 }
?>
 </body>
</html>
```

ADD_MAC.SH

```
#!/bin/bash

# $1 == MAC $2 == USER_NAME

PATH="/usr/local/bin:/usr/bin:/bin"

# sprawdzamy czy taki MAC nie został już zarejestrowany
MAC_LINE=`egrep "^$1[\t ]" /home/*/.dhcp_mac /etc/dhcp3/dhcpd.conf`
# grupa do której należy użytkownik
GID=`egrep "^$2:" /etc/passwd | cut -f4 -d':'`
# sciezka do pliku z MAC'ami uzytkownika
MAC_FILE="/home/$2/.dhcp_mac"
# sciezki do wszystkich plików z MAC'ami grup[y
MAC_GROUP_FILES=`egrep '[^:]*:[^:]*:[^:]*:'$GID':' /etc/passwd | cut -f6 -d':' | while read f;
do echo "$f/.dhcp_mac"; done`

is_OK() {
 if [ $1 = 0 ]; then
  echo " done";
 else
  echo " ERROR";
  exit
 fi
}

if [ "$MAC_LINE" = "" ]; then
 # ustalamy numer kolejny komputera ...
 MAX=`cat $MAC_GROUP_FILES | awk '{printf("%s\n", $2)}' | sort -n | tail -n 1`
if  [ "$MAX" = "" ]; then
  MAX=20;
 elif [ $MAX -gt 253 ]; then
  echo "Przekroczono maksymalną liczbę zarejestrowanych numerów IP dla tego lokalu"
  exit
 fi
 MAX=$(( $MAX + 1 ));

# zapisujemy nowy MAC
 echo -n "Dopisywanie: $1 $MAX do $MAC_FILE ..."
echo -e "\n$1 $MAX" 1>> $MAC_FILE
 is_OK $?

# ustawiamy prawa do pliku (konieczne gdy został utworzony nowy)
 chown "$2:$GID" "$MAC_FILE"

# wywolujemy przbudowanie bazy MAC'adresów i reload dhcp
 echo "Proszę czekać trwa przeładowywanie ustawień DHCP"
/usr/local/bin/dhcp_user_conf.sh
echo "Ustawienia DHCP przeładowane, proszę odnowić adres IP w kliencie"
else
 echo "Podany MAC adres był już zarjestrowany."
fi
```

INSTALL_IN_USERMIN.SH

```
#!/bin/bash
```

```
# funkcja edytująca plik konfiguracyjny
edit_config() {
 if grep "$2" "$1" >/dev/null; then
  tmp_file="/tmp/${PPID}_`basename "$1"`"
  mv "$1" "$tmp_file"
# modyfikacja (sed) i usuwanie nadmiaru pustych linii (awk)
  sed "s#$2#$3#g" "$tmp_file" | awk '
   $0 != "" {
    licz=0
    print $0
    next}
{licz++}
   licz>=3 {next}
{print $0}
  ' > "$1"
  rm "$tmp_file";
 else
  echo "" >> "$1"
  echo "$3" >> "$1"
fi
}

# jeżeli istnieje usermin to preparujemy odpowiedni moduł do edycji listy mac-adresów
# stworzone dla usermin 1.340
if [ -d /usr/share/usermin/plan ]; then
 echo "user: changepass cshrc plan mac_list shell proc ssh procmail spam language cron ⤸
 sip_register sip extension" \
  > /etc/usermin/webmin.acl

cp -r /usr/share/usermin/plan /usr/share/usermin/mac_list

edit_config "/usr/share/usermin/mac_list/plan-lib.pl" '^[$]plan_file.*$' '$plan_file = ⤸
 "$remote_user_info[7]/.dhcp_mac";'

echo "longdesc=Edit your .dhcp_mac file, sent in response to finger requests." > ⤸
 /usr/share/usermin/mac_list/module.info
echo "desc=MAC File" >> /usr/share/usermin/mac_list/module.info
echo "usermin=1" >> /usr/share/usermin/mac_list/module.info
echo "name=MAC" >> /usr/share/usermin/mac_list/module.info

edit_config "/usr/share/usermin/lang_list.txt" '^lang=en,.*$' 'lang=en,charset=UTF-8,⤸
 titles=1        English'
rm /usr/share/usermin/mac_list/lang/*
 echo 'index_title=Plik .dhcp_mac' > /usr/share/usermin/mac_list/lang/en
echo 'index_desc=.dhcp_mac jest to plik z listą MAC adresów i przypisywanymi do nich ⤸
 koncówkami numerów IP.
  Aby wprowadzone zmiany odniosły natychmiastowy skutek należy w przejśc do
  <a href="http://FIXME-system.dodawania.mac.adresow-FIXME/add_mac.php?action=reload">tej ⤸
  strony</a>,
  wykonać ponowne logowanie i poczekać na załadowanie strony.' > ⤸
  /usr/share/usermin/mac_list/lang/en
iconv -f UTF8 -t latin2 < /usr/share/usermin/mac_list/lang/en > ⤸
 /usr/share/usermin/mac_list/lang/pl
fi
```

Asterisk

Popularna w systemach VoIP konfiguracja przez menu telefoniczne jednak nie jest zbyt wygodna i chyba warto umożliwić użytkownikom konfigurację swoich usług poprzez odpowiednie pliki na serwerze - zamieszczam skrypt generujący zbiorczy plik konfiguracyjny dla Asteriska w oparciu o pliki zapisane w katalogach użytkowników (przykładowy konfig). System, automatycznie dodaje do nazw kontekstów identyfikatory użytkowników do których one należą (dzięki czemu nie ma się możliwości grzebania w cudzej konfiguracji), działanie to oparte jest na założeniu że identyfikator odpowiadający użytkownikowi shellowemu jest w prosty sposób zależny od jego UID. System także importuje lub usuwa odpowiednie wpisy bazy danych należące do użytkownika (uwagi i założenia jak poprzednio. A następnie dokonuje zrzutu wpisów bazy danych dotyczących użytkownika do pliku w jego katalogu domowym. Ponadto zamieszczam program umożliwiający dzwonienie do samego siebie w oparciu o mechanizm ”auto-dial out call files” (więcej informacji w komentarzu wprowadzającym).

Konieczne jest aby program callme należał do użytkownika na prawch którego działa serwer asteriska oraz maił ustawiony SUID i GUID (`chown asterisk:asterisk /usr/local/bin/callme && chmod 6755 /usr/local/bin/callme`). Dla wygodnego korzystania z systemu należy dodać do `/etc/sudoers` następującą linię:

```
ALL ALL =NOPASSWD: /usr/local/bin/asterisk_user_conf.sh
```

Uwaga: prezentowane powyżej programy mają status beta - przechodzą one dopiero testy i mogą pojawić się jakieś błędy.

ASTERISK_USER_CONF.SH

```
#!/bin/bash

# skrypt generujący zbiorczą konfigurację uzytkowników asteriska w poarciu o pliki w ⤸
katalogach użytkowników

# * podstawowy numer telefonu przydzielony użytkownikowi odpowiada jego UID'owi ⤸
pomniejszonemu o $MIN_UID
#     i poprzedzonemu prefixem UID_NUM_PREF
# * nie przydziela się kont SIP z numerami w oparciu o GID
#     ze względu na brak konta shellowego które przechowywałoby konfiguracje ...
# * numery oparte o GID (numery wspólne grup urzytkowników) można tworzyć jako aliasy,
#     numery te powinny mieć inny prefix niż numey tworzone w oparciu o UID oraz niż ⤸
normalne aliasy

PATH="/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/bin"

USER_REGISTER='/var/user_config/sip-register.conf'
USER_SIP_CONF='/var/user_config/sip.conf'
USER_DIALPLAN='/var/user_config/extensions.conf'
USER_ACCOUNT='/var/user_config/auto-account.conf'
USER_ACCOUNT_VOICEMAIL='/var/user_config/auto-account-voicemail.conf'
MAIL_DOMAIN=`cat /etc/mailname`
MIN_UID=1000
UID_NUM_PREF="4"

make_section_for_user() {
# wybiera podstawie pliku $3 (lub STDIN gdy niepodano) sekcje określoną przez $2
# do nazwy sekcji dodaje identyfikator właściciela określony przez $1
# $2 jest wyrażeniem regularnym zgodnym z notacją stosowaną w awk
 grep -v '$[ \t]*[;#]' "$3" | awk -v UID=$1 -v SECTION=$2 '
  BEGIN {
   RS="(^|[\n])[\t ]*[[]"
   FS="[]][ \t]*[\n]"
  }

  $1 ~ SECTION {
```

```
    sectionsubname=gensub("KONTO-?", "", 1, $1)
    if (sectionsubname == "")
     printf("[%s]\n%s\n", UID, $2)
    else
     printf("[%s-%s]\n%s\n", UID, sectionsubname, $2)

    # w sekcjach dial-out musimy dodać powrót w przypadku invalid
    # wymagane przez GoSub()
    if ($1 == "dial" && sectionsubname == "out")
     printf(" exten => i,1,Return\n")
   }
  ' | sed -e "s#KONTO#$1#g"
}


# konfiguracja bazy danych
for f in /home/*/.asterisk.db.in; do
 if [ -e "$f" ]; then
  # uzyskujemy identyfikator użytkownika
  #SIP_UID=$(id -u $(basename `dirname /home/rrp/.asterisk.cfg`))
  SIP_UID=$UID_NUM_PREF$(( $(id -u `ls -ld $f | cut -f3 -d' '`) - $MIN_UID ))

  mv $f $f.tmp
  while read key data; do
   if [ $key !="" ]; then
    if [ "$data" = "" ]; then
     asterisk -rx "database deltree $SIP_UID $key"
    else
     asterisk -rx "database put $SIP_UID $key $data"
    fi
   fi
  done < "$f"
  rm $f.tmp
fi
done

# plik powinien być postaci: każda linia jeden wpis
# pierwsze pole w linii klucz (bez identyfikatora użytkownika),
# drugie (odzielone białymi znakami) dodawane do bazy danych dane
# gdy drugie puste to kasujemy drzewo podane w key

# konfiguracja z plików tekstowych i zrzut bazy danych
:> $USER_REGISTER
:> $USER_SIP_CONF
:> $USER_DIALPLAN
echo "[sip-dial-in]" >> ${USER_DIALPLAN}-1
for f in /home/*/.asterisk.cfg; do
 # uzyskujemy identyfikator użytkownika
 SIP_UID=$UID_NUM_PREF$(( $(id -u `ls -ld $f | cut -f3 -d' '`) - $MIN_UID ))

# wybieramy register
 register=`make_section_for_user "$SIP_UID" '^register$' "$f"`

# zapisujemy zmodyfikowany register
 echo "$register" | egrep '^[ \t]*register[ \t]*=>' | while read reg; do
  ext=`echo $reg | awk 'BEGIN {FS="[>:@/]"} {printf("%s-%s", $4, $2)}' | tr -d ' '`
  org_ext=`echo $reg | cut -f2 -s -d'/'`
  reg=`echo $reg | cut -f1 -d'/'`
  echo "$reg/$ext" >> "$USER_REGISTER"
  if [ $org_ext ]; then
```

```
    echo " exten => $ext,1,Goto($SIP_UID-dial-in,$org_ext,1)" >> ${USER_DIALPLAN}-1
   else
    echo " exten => $ext,1,Goto($SIP_UID-dial-in,s,1)" >> ${USER_DIALPLAN}-1
   fi
  done;

 # wybieramy sekcję związane z konfiguracją provider'ów SIP
  make_section_for_user "$SIP_UID" '^sipprovider-.*' "$f" |
   sed -e 's#[ \t]*context[ =].*$# context = sip-dial-in#g' >> $USER_SIP_CONF

 # wybieramy sekcję związane z konfiguracją SIP
  make_section_for_user "$SIP_UID" '^KONTO.*' "$f" >> $USER_SIP_CONF

 # wybieramy sekcję związane z konfiguracją kontekstów planu numeracyjnego
  make_section_for_user "$SIP_UID" '^dial-.*' "$f" >> ${USER_DIALPLAN}-2

 asterisk -rx "database show $SIP_UID" > `dirname $f`/.asterisk.db.dump
done

echo "" >> ${USER_DIALPLAN}-1
cat ${USER_DIALPLAN}-1 ${USER_DIALPLAN}-2 > ${USER_DIALPLAN}
rm -f ${USER_DIALPLAN}-1 ${USER_DIALPLAN}-2

# pliki użytkowników zawierają konteksty
# * [register] z wpisami register => dotyczącymi użytkownika
#    (jezeli podano /ext przychodzaca rozmowa trafi do ext,1,... w [dial-in])
# * [sipprovider-$nazwa] z konfiguracją dzwonienia przez providera $nazwa
# * [dial-in], z konfiguracją połączeń przychodzących od providerów SIP
#    (rozmowy na adresy standardowe SIP - loginy, UIDy, aliasy, ...
#     kierowane są domyślnie do int,1 w [dial-in])
# * [KONTO] lub [KONTO-$subid], z konfiguracją dla swoich telefonów IP itp
#    (konteksy wyjsciowe dla rozmow nalezy podawac jako KONTO-dial-out lub KONTO-dial-$nazwa-
out)
# * [dial-out] lub [dial-$nazwa-out], z konfiguracją połączeń wychodzących z telefonu
#    (nazwa kontekstu musi odpowiadać podanej w konfiguracji konta telefonu)
#    (dzwonienie należy realizować na adresy @KONTO-sipprovider-$nazwa)
# słowo kluczowe KONTO jest automatycznie zamieniane na nazwę konta

# generowanie pliku automatycznych kont w oparciu o /etc/passwd
:> $USER_ACCOUNT
:> $USER_ACCOUNT_VOICEMAIL
while IFS=":" read name tmp1 uid tmp2 rname home; do
 if [ $MIN_UID -lt $uid -a "$name" != "nobody" ]; then
  SIP_UID=$UID_NUM_PREF$(( $uid - $MIN_UID ))
  echo " exten => $name,1,GoTo($SIP_UID,1)" >> $USER_ACCOUNT
  echo " exten => $SIP_UID,1,GoTo($SIP_UID-dial-in,int,1)" >> $USER_ACCOUNT
  if [ -e $home/.asterisk.cfg ]; then
   password=`make_section_for_user "$SIP_UID" '^general' "$home/.asterisk.cfg" |
    awk 'BEGIN {FS="=>"} $1 == "password" {print $2}'`
  else
   password="$RANDOM"
  fi
  echo " $SIP_UID => $RANDOM,$rname,$name@$MAIL_DOMAIN,,delete=1" >> $USER_ACCOUNT_VOICEMAIL
 fi
done < /etc/passwd

chown asterisk:asterisk $USER_REGISTER $USER_SIP_CONF $USER_DIALPLAN $USER_ACCOUNT
$USER_ACCOUNT_VOICEMAIL
chmod 660 $USER_REGISTER $USER_SIP_CONF $USER_DIALPLAN $USER_ACCOUNT $USER_ACCOUNT_VOICEMAIL
```

```
# przeładowanie konfiguracji
asterisk -rx "reload" > /dev/null
```

USER-ASTERISK.SAMPLE.CONF

```
; zastąpić:  USER_NAME1 USER_NAME2 PASSWORD1 PASSWORD2 PASSWORD3 PASSWORD4 PASSWORD5
;            sip.provider.example.com provider.example.com REAL_NAME REAL_PHONE
; swoimi danymi konfiguracyjnymi
; nie zastępować KONTO (jest podmieniane automatycznie na identyfikator konta)
; więcej szczegółów w asterisk_user_conf.sh

[general]
; hasło do systemu konfiguracyjnego i poczty głosowej
password => PASSWORD5

[register]
register => USER_NAME1:PASSWORD1@sip.provider.example.com
register => USER_NAME2:PASSWORD2@sip.provider.example.com/thom

[dial-in]
exten => s,1,Set(CALLERID(all)="${CALLERID(name)} @ PROVIDER_NAME" <00${CALLERID(num)}>)
exten => s,n,Macro(number_classification||KONTO||anonymous|no|$["${CHANNEL}" : "([A-Za-z0-9]+⤸
)/"])
exten => s,n,Hangup

exten => thom,1,Set(CALLERID(all)="${CALLERID(name)} @ PROVIDER_NAME" <00${CALLERID(num)}>)
exten => thom,n,Set(__SUB_EXT=thom)
exten => thom,n,Macro(number_classification||KONTO||anonymous|no|$["${CHANNEL}" : "([A-Za-z0-⤸
9]+)/"])
exten => thom,n,Hangup
exten => thom,100,Macro(call_user|KONTO|SIP/KONTO-thom|||)
; UWAGA: jezli jakies argumenty sa opcjonalne nalezy je podawac jako puste
;        inaczej moga trafic argumenty makra nadrzednego (bug ??)

exten => int,1,GotoIf($["${SUB_EXT}" != "" ]?${SUB_EXT},100)
; wysyłanie powiadomienia o przychodzącym połączeniu przez XMPP
; wymaga aby w jabber.conf był skonfigurowany dostęp do konta jabbera
; z którego będzie wysłane powiadomienie - w sekcji [asterisk]
; (nazwa sekcji zgodna z 1 argumentem poniższego wywołania)
exten => int,n,jabbersend(asterisk|my_jid@example.net|${STRFTIME(,GMT,%F %T)} UTC: dzwnoni ${⤸
CALLERID(all)})
;exten => int,n,Macro(call_user|KONTO|SIP/KONTO-tel|15|SIP/KONTO-thom&SIP/KONTO-tel|30|)
;exten => int,n,Macro(call_user|KONTO|SIP/KONTO-tel|150|||)
  exten => int,n,Macro(call_user|KONTO|SIP/KONTO-tel&SIP/KONTO-thom|150|||)

; obsługa dzwonienia do samego siebie z asteriska celem powiadomienia o nowym mailu ...
exten => new_mail,1,Answer
exten => new_mail,n(czytaj),Playback(wprowadzenie.wav)
exten => new_mail,n,Playback(${komunikat})
exten => new_mail,n,Read(MODE|nacisnij_aby_powtorzyc.wav|1)
exten => new_mail,n,GotoIf($["${MODE}" != ""]?czytaj)
exten => new_mail,n,HangUp()


[dial-out]
; powrót dla Gosub z redirect i transfers
exten => i,1,Set(GO_IF_NOT_FOUND=1)
exten => i,2,Return
```

```
; numery specjalne
exten => 1,1,NoOp
exten => 1,n,Macro(call_user|KONTO|SIP/KONTO-thom|150|||)
exten => 1,n,Hangup

; prefiksowe wyjście na zewnątrz przez operatora SIP/IAX
exten => _0X.,1,Macro(number_classification||${EXTEN}|${SIPDOMAIN}||$[ 1 + ${PRIORITY} ]|$[
"${CHANNEL}" : "([A-Za-z0-9]+)/"])
exten => _0X.,n,Verbose("OUT via rrp-PROVIDER_NAME")
exten => _0X.,n,Set(CALLERID(all)="Robert Paciorek" <048222443895}>)
exten => _0X.,n,Dial(SIP/${EXTEN:1}@KONTO-sipprovider-PROVIDER_NAME||${
DIAL_OPTIONS_TO_NON_LOCAL})
exten => _0X.,n,GotoIf($["${DIALSTATUS}" = "BUSY"]?busy:hangup)
exten => _0X.,n(busy),Busy
exten => _0X.,n(hangup),Hangup

; dzwonienie na inne numery / URI
exten => _[1-9a-zA-Z.-*#+][0-9a-zA-Z.-*#+]!,1,Macro(number_classification||${EXTEN}|${
SIPDOMAIN}||no|$["${CHANNEL}" : "([A-Za-z0-9]+)/"])
exten => _[0-9a-zA-Z.-*#+][a-zA-Z.-*#+][0-9a-zA-Z.-*#+]!,1,Macro(number_classification||${
EXTEN}|${SIPDOMAIN}||no|$["${CHANNEL}" : "([A-Za-z0-9]+)/"])

[dial-DIAL_NAME-out]
exten => 1,1,Set(CALLERID(all)="matka" <1>)
exten => 1,n,Macro(call_user|KONTO|SIP/KONTO-tel|150|||)
exten => 1,n,Hangup

exten => _0X.,1,Macro(number_classification||${EXTEN}|${SIPDOMAIN}||$[ 1 + ${PRIORITY} ]|$[
"${CHANNEL}" : "([A-Za-z0-9]+)/"])
exten => _0X.,n,Dial(SIP/${EXTEN:1}@KONTO-sipprovider-PROVIDER_NAME2||${
DIAL_OPTIONS_TO_NON_LOCAL})
exten => _0X.,n,GotoIf($["${DIALSTATUS}" = "BUSY"]?busy:hangup)
exten => _0X.,n(busy),Busy
exten => _0X.,n(hangup),Hangup

[sipprovider-PROVIDER_NAME2]
type=friend
host=sip.provider.example.com
username=USER_NAME2
secret=PASSWORD2
fromuser=USER_NAME2
fromdomain=provider.example.com
insecure=invite
context=rrp-PROVIDER_NAME-in
disallow=all
allow=g726
allow=g729
[sipprovider-PROVIDER_NAME]
type=peer
host=sip.provider.example.com
username=USER_NAME1
secret=PASSWORD1
fromuser=USER_NAME1
fromdomain=provider.example.com
insecure=invite
context=rrp-PROVIDER_NAME-in
disallow=all
allow=g726
allow=g729
```

```
[KONTO-tel]
type = friend
host = dynamic
callerid = REAL_NAME <REAL_PHONE>
secret = PASSWORD3
context = KONTO-dial-out
[KONTO-thom]
type = user
host = dynamic
insecure=invite
callerid = REAL_NAME <REAL_PHONE>
context = KONTO-dial-DIAL_NAME-out
[KONTO-thom]
type = peer
host = dynamic
secret = PASSWORD4
```

CALLME.C

```
/** program służy do przygotowywania i umieszczania w odpowiednim katalogu plików
 *   wywołujących dzwonienie od asteriska
 *
 *   program powinien chodzić z SUID użytkownika na którym działa asterisk
 *   (prawa do plików w tym katalogu są restrykcyjnie sprawdzane, ponadto
 *   katalog nie powinien być globalnie zapisywalny)
 *
 *   program umożliwia dzwonienie do samego siebie w systemie kont opartych
 *   o UIDy użytkowników systemowych (opisanym szerzej w
 *   "Konfiguracja w ręce użytkowników") i może być wykorzystany np.
 *   do powiadamiania o nowej poczcie
 *
 *   wymaga to wcześniejszego przygotowania pliku dźwiękowego (np. przy pomocy
 *   text2wave) w oparciu o informacje o nowej poczcie (np. w wyniku przetwarzania
 *   ~/.procmailrc) oraz wykonania tego programu z argumentem będącym ścieżką do tego
 pliku
 *
 *   ponadto wymaga istnienia odpowiedniego kontekstu w pliku konfiguracyjnym asteriska
 *   (w ~/.asteris.cfg w [dial-in] extension new_mail zgodny z podanym w przykładzie)
 *
 *   użytkownik musi samodzielnie zadbać o usunięcie pliku dźwiękowego
 *   po odpowiednim czasie (np. po 0.5h)
 *
 *   działanie programu odpowiada mniej więcej wywołaniu
 *   asterisk -rx 'originate SIP/41-tel extension new_mail@user-dial-in'
 *   tyle że użytkownik może dzwonić tylko do siebie i wskazać plik do odtworzenia
 *
 *   Więcej informacji o wykorzystanym mechanizmie w
 *   http://www.voip-info.org/wiki-Asterisk+auto-dial+out
**/

#include <stdio.h>
int main (int argc, char *argv[]) {
 char fnameA[100], fnameB[130];
 int pid, uid, uid_prefix = 4;
```

```c
pid = getpid();
 uid = getuid();

sprintf(fnameA, "/var/tmp/callme_%d_%d.txt", uid, pid);
 sprintf(fnameB, "/var/spool/asterisk/outgoing/callme_%d_%d.txt", uid, pid);

uid = uid - 1000;

FILE *plik;
 plik=fopen(fnameA, "w+");
 if (plik == NULL) {
  perror("Blad otwarcia pliku:");
  return -1;
 }

fprintf(plik, "Channel: Local/%d%d\n", uid_prefix, uid);
 fprintf(plik, "CallerID: New Mail <NewMail>\n");
 fprintf(plik, "MaxRetries: 2\n");
 fprintf(plik, "RetryTime: 120\n");
 fprintf(plik, "WaitTime: 45\n");
 fprintf(plik, "Context: %d%d-dial-in\n", uid_prefix, uid);
 fprintf(plik, "Extension: new_mail\n");
 if (argc > 1)
  fprintf(plik, "Set: komunikat=%s\n", argv[1]);

close(plik);
 if ( rename(fnameA, fnameB) ) {
  perror("Blad przenoszenia pliku:");
  return -1;
 }
}
```

Porady konfiguracyjne

Dział ten gromadzi porady konfiguracyjne związane z różnymi systemami POSIXowymi. Omawiane są w nim zagadnienia ”systemowe”, porady związane z specyficznymi zastosowaniami (takimi jak usługi sieciowe, multimedia, ...) opisane są w innych artykułach w ramach działu ”użytkowanie i konfiguracja”, a niekiedy także w artykule ”Oprogramowanie użytkowe”.

Poradnik linuxowy - podstawy konfiguracyjne

Klawiatura

Uwagi ogólne dotyczących konfigurowania klawiatury tak pod konsolą jak i pod X’ami:

- mapę klawiszy pod konsolą ustawiamy poleceniem `loadkeys /usr/share/keymaps/i386/qwerty/pl.kmap.gz`, za domyślną mapę klawiatury odpowiada `/etc/console/boottime.kmap.gz` jest ona wczytywana przez skrypt startowy `/etc/init.d/keymap.sh`
- mapę klawiszy w X’ach ustawiamy poleceniem `setxkbmap -model pc104 -layout pl -variant basic`, za ustawienia domyslne odpowiadają wpisy w `/etc/X11/xorg.conf`, istotne są także ustawienia używanego środowiska takiego jak KDE czy gnome

— za wybór układu fizycznej klawiatury pod konsolą odpowiadają ustawienia w /proc (dokonywane przez /etc/init.d/keymap.sh), pod X'ami są to wpisy w pliku konfiguracyjnym i opcje setxkbmap

Należy także rozróżniać ustawienia klawiatury, od ustawienia czcionek, od ustawienia locali. Ustawienia klawiatury (opisane powyżej odpowiadają za to jakie znaki jesteśmy wstanie wprowadzić), ustawienia czionek (pod konsolą wpis SCREEN_FONT=iso02.f16 w /etc/console-tools/config) odpowiadają za to jakie znaki system jest w stanie wyświetlić. Natomiast ustawienia locale (zmienne LANG i LC_*, ustawienie ogólnosystemowe w /etc/environment) odpowiadają za to w jakim języku "rozwmawia" z nami większość programów, w jaki sposób zakodowane są znaki non-asci w tych rozmowach, oraz za domyślne kodowanie w wielu edytorach tekstowych.

Oprócz typowego wprowadzania znaków niedostępnych bezpośrednio (metoda ta nie ogranicza się tylko do polskich ogonków) z użyciem modyfikatora Alt-Gr (prawy Alt) możliwe jest także wprowadzania niedostępnych znaków przy pomocy uzyskiwanego przez niektóre z kombinacji Alt-Gr martwego klawisza[1772] oraz klawisza komponującego[1773]. Aby skorzystać z klawisza komponującego konieczne jest przypisanie go do ktyóregoś z modyfikatorów na klawiaturze np. do prawego WinKey - można to zrobić poprzez setxkbmap -option compose:rwin lub rekonfigurację pakietu "keyboard-configuration". Klawisz komponujący potwrafi używać indywidualnej mapy dla użytkownika umieszczonej w pliku ~/.XCompose (za wzorzec mogą posłuyzyć mapy systemowe z /usr/share/X11/locale/*/Compose). W ostateczności można także znak przekopiować z jakiejś tablicy znaków Unicode. Zobacz także: Alt-kody[1774], Wprowadzanie znaków niedostępnych na klawiaturze[1775]

Klawiatura numeryczna i kropka

Aby uzyskamy kropkę zamiast przecinka na klawiaturze numerycznej należy wykomentować w pliku /usr/share/X11/xkb/symbols/pl linijkę include "kpdl(comma)" (dawniej include "keypad(comma)") w sekcji odpowiadającej naszej klawiaturze (dla "polski programisty" sekcja "basic").

W środowisku graficznym taki efekt możemy także uzuskać komendą: setxkbmap pl; xmodmap -e "keycode 91 = KP_Delete period KP_Delete period". Program ten może także posłużyć do innych modyfikacji klawiatury np. xmodmap -e "clear mod1" -e "clear mod3" -e "clear mod4"; xmodmap -e "add mod1 = Alt_L" -e "add mod3 = Super_R" -e "add mod4 = Super_L".

Klawiatura multimedialna i nie tylko

W przypadku konsoli po wciśnięciu jakiegoś z dodatkowych przycisków zobaczymy komunikat o nieznanym numerze klawisza możemy wtedy przy pomocy programu setkeycodes przypisać mu konwencjonalny numer i następnie wykorzystać go w mapie klawiatury. W przypadku klawiszy mających już przypisany konwencjonalny numer możemy go poznać za pomocą komendy showkey.

W przypadku X'ów możemy skorzystać z programów pozwalających na przypisanie tym klawiszom komendy (takich jak "hotkeys" czy "xbindkeys"). Możemy jednak przypisać im normalne oznaczenia funkcyjne poprzez xmodmap i keycode (przykładowy plik konfiguracyjny .xmodmaprc[1776]). Kody klawiszy zczytać możemy przy wykorzystaniu programu xev, natomiast listę nazw do których mogą być przypisane można znaleźć w pliku /usr/lib/X11/XKeysymDB. Następnie po wykonaniu xmodmap ~/.xmodmaprc (warto wpisać jako w autostarcie) możemy tych klawiszy używać jak każdych innych - w ustawieniach skrótów klawiaturowych w aplikacjach, w konfiguracji menu KDE.

Klawisze te możemy w szczególności wykorzystać w dość rozbudowanym systemie akcji klawiaturowych oferowanym przez KDE. Znajdziemy go w "regionalne i dostępność" -> "Input Action" (jeden z znajdujących się tam przykładów pokazuje jak zrobić aby skrót klawiaturowy wpisywał podany tekst, oprócz tego możemy definiować skróty wieloklawiszowe (specjalny + coś i kolejne stanowiące napis ...).

Odbiornik IR i sterowanie pilotem

Możliwe jest także korzystanie z pilotów podczerweni do sterowania komputerem z linuxem. Wiele z tunerów telewizyjnych DVB posiada wbudowany odbiornik podczerwieni, jeżeli jest on obsługiwany przez nasze jądo mapowania jego przycisków na sygnały klawiatury możemy dokonać z wykorzystaniem ir-keytable. W przypadku popularnego układu RTL2832U opis konfiguracji możemy znaleźć w stosownym artykule na linuxtv.org[1777]. Plik z tablicą mapowań zapisujemy w /etc/rc_keymaps/, testowanie (uzyskiwanie iformacji o identyfikatorze wciśniętego przycisku) przeprowadzamy przy pomocy komendy: ir-keytable -s rc0 -t -v, przeładowanie używanej tablicy przy pomocy ir-keytable -s rc0 -w /etc/rc_keymaps/rtl_simple (uwaga: po dodaniu nowych KEY_XXXX do tablicy aby były zauważone w xev należy zrestatować xserver; wynika to z tego ze xserwer ogranicza sie do obslugi eventow odczytanych na starcie z /dev/input/event* jako obsługiwane przez dane urzadzenie, możemy je

odczytać manualnie z yzyciem polecenia `evtest /dev/input/event12`). Informację na temat mapowania symboli `KEY_XXXX` na symbole uzywane w Xach mozna podejrzec np. w: input_linux_map_fixed_old.txt[1778]

aby ustawienia były wczytywane automatycznie po starcie systemu / podłączeniu urządzenia należy w pliku `/etc/rc_maps.cfg` dodać wpis wiążący moduł obsługujący odbiornik IR z mapą klawiszy - np: `dvb_usb_rtl28xxu * rtl_simple`, gdzie mapa zapisana w pliku `/etc/rc_keymaps/rtl_simple` ma postać:

```
# table rtl_simple, type: NEC
# duzy pilot - srodkowe "kolko" (L, R, ENTER, UP, DOWN)
0xe5004c KEY_PREVIOUSSONG
0xe50040 KEY_NEXTSONG
0xe50006 KEY_PLAYPAUSE
0xe50044 KEY_VOLUMEUP
0xe50048 KEY_VOLUMEDOWN
# duzy pilot - numeryczne (1, 3, 4, 6, 7, 9) jako jump
0xe50009 KEY_BACK
0xe50001 KEY_FORWARD
0xe5004b KEY_PAGEUP
0xe50043 KEY_PAGEDOWN
0xe5004a KEY_HOME
0xe50042 KEY_END
# inne
0xe50057 KEY_PLAYPAUSE
0xe5005e KEY_PLAYPAUSE
0xe5005a KEY_STOPCD
```

Zaawansowana konsola - framebuffer i mysz

Istnieje możliwość uruchomienia konsoli linuxowej w trybie "graficznym". Umożliwia to wyświetlanie plików graficznych, oglądanie telewizji, granie w niektóre gry, jak również wyświetlanie dużo większej liczby linii tekstu. Aby uruchomić system z taką konsolą konieczne jest dodanie do parametrów startowych jądra parametru `vga=XXX`, gdzie XXX jest zapisanym szesnastkowo (dla GRUB) lub dziesiętnie (dla LILO) kodem danego trybu. Poniżej tabelka dostępnych kodów (w postaci szesnastkowej, zaczerpnięta z linux/Documentation/fb/vesafb.txt):

```
    | 640x480  800x600  1024x768 1280x1024
----+-------------------------------------
256 |  0x301    0x303    0x305    0x307
32k |  0x310    0x313    0x316    0x319
64k |  0x311    0x314    0x317    0x31A
16M |  0x312    0x315    0x318    0x31B
```

Kożystanie z framebuffera może prowadzić do pewnych problemów (o czym poniżej). Najczęściej wykorzystywany jest standardowy sterownik vesafb istnieją jednak moduły dla poszczególnych kart graficznych. Istnieje też interesująca biblioteka - DirectFB obsługująca wyświetlanie w trybie framebuffera, w przypadku kompilacji z opcją –enable-multi umożliwia pracę kilku aplikacji (może robic za coś na kształt menagera okien), szczegóły w http://www.directfb.org/downloads/Core/DirectFB-1.0/README

Przydatną rzeczą na konsoli jest również myszka. Aby móc z niej korzystać należy zainstalować pakiet gpm , który umożliwia uzycie komputerowego gryzonia na konsoli oraz kopiowanie i wklejanie przy jego pomocy tekstu. Konfiguracja tak aby wszystko działało dobrze (zwłaszcza gdy chcemy mieć też X'y) może przysporzyć trochę zabawy. W moim wypadku (radiowa myszka z rolką) największym problemem było dobranie rodzaju myszki w X'ach aby ta nie była nadwrażliwa ... wcześniejsze imps2 nie chciało poprawnie działać z /dev/gpmdata, w którym kopie danych z myszki umieszcza gpm; działająca konfiguracja wygląda natomiast tak:

`/etc/gpm.conf`:

```
# dla PS/2:
device=/dev/psaux
# dla USB (wymaga modolow hid i usbmouse):
#device=/dev/input/mice
# lub:
#device=/dev/input/mouse0
type=autops2
#type=imps2
```

```
responsiveness=
repeat_type=ms3
append=""
sample_rate=
```

/etc/X11/XF86Config-4:

```
Section "InputDevice"
        Identifier      "Configured Mouse"
        Driver          "mouse"
        Option          "CorePointer"
        Option          "Device"                "/dev/gpmdata"
        Option          "Protocol"              "IntelliMouse"
        Option          "ZAxisMapping"          "4 5"
EndSection
```

Może się zdażyć że aby gpm działało wymaga dwu-trzykrotnego uruchomienia ...

Jeżli na jednej z tak uzyskanych pięknych konsoli chielibyśmy mieć zamiast lokalnego logowania logowanie zdalne po ssh możemy skozystać z poniższego wrappera umożiwiającego odpalanie takiego połączenia z getty:

```
#!/usr/bin/perl
# put in /etc/inittab:
# 2:23:respawn:/sbin/getty -ni -l /usr/local/ssh_to_xxx 38400 tty2
# where /usr/local/ssh_to_xxx is path to this script with correct $hostname variable:
my $hostname="baszta.vls.icm.edu.pl";
print "$hostname login: ";
my $username = <STDIN>;
chomp ( $username );
exec ("/usr/bin/ssh $username\@$hostname");
```

Sortowanie w polskich localach

Domyślna konfiguracja sortowania w polskich (i nie tylko) lokalizacjach jest taka że występujące na początku nazw plików znaki takie jak kropki, wykrzykniki, małpy itp. są ignorowane. Mnie to się nie podobało i postanowiłem to zmienić: odnalazłem plik źródłowy dla pl_PL - `/usr/share/i18n/locales/pl_PL` i przerobiłem sortowanie na (w mojej opinii) bardziej normalne. Tutaj[1779] zamieszczam zmodyfikowany plik źródłowy należy go wrzucić do podanej lokalizacji (lepiej wcześniej zrobić backup tego co tam jest) i przebudować locale - np. przez rekonfigurację stosownego pakietu.

Network File System

NFS[1780] jest opartym o protokół IP (UDP lub TCP) protokołem do udostępniania systemów plików poprzez sieć. Udostępnianie zasobów dyskowych po nfs wymaga zainstalowanego serwera NFS (dla linuxów występuje on w dwóch wersjach - działającej w przestrzeni jądra oraz działającej jako normalny daemon). Konfiguracja udostępnianych zasobów ustawiana jest w `/etc/exports` (dla linuxa) i zatwierdzana (przeładowywana) przy pomocy komendy exportfs (opcja -r dla ponownego eksportu - przeładowania configu, -v dla pokazania co jest obecnie wyeksportowane). Plik składa się z dwóch kolumn - w pierwszej podawana jest ścieżka do udostępnianego systemu plików, w drugiej podawane są adresy uprawnionych hostów i opcje exportu dla każdego z hostów. Konfiguracja montowania odbywa się standardowo w `/etc/fstab` i wymaga podania jako typu systemu plików `nfs`.

Możliwe jest także eksportowanie drzewa katalogów po nfs-ie bez umieszczania wpisów w plikach konfiguracyjnych - `exportfs -o rw lub rohost-klienta:/eksportowana/sciezka` (w niektórych -nie linuxowych - systemach `share -o rw=host-klienta-z-prawem-zapisu,ro=host-klienta-tylko-do-odczytu/eksportowana/sciezka` oraz montowanie - `mount -t nfs serwer:/eksportowana/sciezka/punkt/montowania`

multipath i SAN

Multipath może być uzyty do konfioguracji ścieżek dostepu do dysków/macierzy dyskowych SAN[1781] podłączanych naprzykład poprzez Fibre Channel[1782]. Na ogół jest narzędziem na tyle sprytnym iż radzi sobie z autodetekcją poprawnej ścieżki itd, oczywiście istnieje także możlwiość manualnej konfiguracji ścieżek.

Multipath pokazuje identyfikatory Fibre Channel[1783] LUN - konkretnie "tgt_node_name" (np. 0x500a09808657d365) oraz "h:b:t:l" (np. 0:0:2:5), gdzie "h:b:t" to `${target_id}`, "tgt_port_name", "port_id" i inne można zobaczyć w `/sys/class/fc_transport/target${target_id}/port_name` i jemu podobnych. Z koleji identyfikator WWN karty (interfejsu) FC (potrzebny do wystawienia zasobów z macierzy) możemy odczytać przez: `cat /sys/class/fc_host/host0/port_name`.

Uwaga: włączenie multipath może spowodować jego zadziałanie na dysku lokalnym co objawi się np. komunikatem o zajętości urządzenia przy próbie tworzenia systemu plików na nim (mkfs mówi "is apparently in use by the system; will not make a filesystem here") - należy wtedy (jeżeli nie jest nam potrzebne) wyłączyć działanie multipath na takim urządzeniu lub operować na urządzeniu multipathowym.

Aby zabronić odpalania się multipath na konkretnym urządzeniu należy w pliku `/etc/multipath.conf` umieścić wpis postaci:

```
blacklist {
        wwid NASZ_WWID
}
```

gdzie `NASZ_WWID` można uzyskać z wyniku komendy `multipath -ll` jest to to identyfikator bez spacji pokazywany w pierwszej linii opisującej dany zastaw ścieżek - należy odrzucić ostatnie znaki tożsame z oznaczeniem urządzenia device mappera w /dev (np. dm-0). Opcjonalnie zamiast wpisu `wwid NASZ_WWID` można podać wpis typu `devnode /dev/sda*`, ale może to rodzić problemy gdy oznaczenia dysków z jakiś powodów się pozmieniają.

Zobacz też inne zagadnienia związane z rozproszonym storagem: iSCSI[1784], Lustre[1785], RAID[1786], NAS[1787], DAS[1788] oraz FreeNAS[1789] - serwer NAS dla FreeBSD

Odświeżanie listy urządzeń SCSI

W przypadku korzystania z urządzeń SCSI hot-plug (np. dysków hot-swap) zachodzi niekiedy potrzeba odświeżenia wykazu urządzeń w systemie, możemy to wykonać w następujący sposób:

```
odświeżenie listy urządzeń (utworzenie plików w /dev, ...)
        echo - - - >/sys/class/scsi_host/HOST/scan  #HOST określa który konroler
↪  poddawany jest ponownemu skanowaniu
```

Często bywa tez potrzebne odświerzenie informacji o wielkości istniejącego urządzenia (np. zmiana rozmiaru LUNa na macierzy). Uzyskać to możemy przy pomocy komend:

```
        # reskan urządzenia (niekiedy wymagane dla ścieżek multipath-owych)
        echo 1 > /sys/block//dev/sda/device/rescan
        # odświeżenie informacji o wielkości urządzenia
        blockdev --rereadpt /dev/sda
        # wielkość możemy sprawdzić komendą (uwaga wynik w blokach 0.5kB):
        blockdev --getsz /dev/sda
        # zmiana rozmiaru urządzenia multipath-owego
        multipathd -k"resize map 360a980006467435a6c5a6441686d5652"
```

Dla sieciowej odmiany SCSI (iscsi) odświeżenie rozmiaru zasobu wykonujemy poleceniem `iscsiadm -m node -R`

Dodatkowo przydatne mogą być polecenia postaci:

```
lsscsi
echo 'scsi-add-single-device    HOST CHAN DEV LUN' > /proc/scsi/scsi
echo 'scsi-remove-single-device HOST CHAN DEV LUN' > /proc/scsi/scsi
```

Kilka sztuczek z /etc/fstab

Plik `/etc/fstab` określa punkty montowań dla poszczególnych urządzeń, a także to które urządzenia (partycje, dyski, ...) będą montowane automatycznie przy starcie systemu.

Plik ten można także (podobnie jak bootloader) konfigurować w oparciu o UUID wartości te najłatwiej odczytać z ls -l `/dev/disk/by-uuid/`). W pliku tym często umieszcza się rzeczy takie jak /proc czy /sys jednak (w odróżnieniu od swap_- `/dev/swap_dev none swap sw 0 0`) nie jest to niezbędne (i tak będą montowane)

W przypadku montowania /var lub /tmp na innej partycji poprzez opcje bind:

```
/home/var           /var       none      bind                0 0
/home/tmp           /tmp       none      bind,umask=000      0 0
```

warto zapewnić kontrolę poprawności ich zamontowania na wypadek problemów z partycją na której są montowane:

```
#!/bin/bash
if ! grep home /proc/mounts > /dev/null; then
        sleep 30
        mount /home
        if ! mount | grep home; then
                echo "Can't mount /home - PLESE FIX THIS"
                env TMOUT=120 bash
        fi
        mount /tmp
        mount /var
fi
```

W tym miejscu warto wsponieć także o możliwości montowania plików jak urządzeń (`dd if=/dev/zero bs=1M count=100 of=/tmp/plik; mkfs.ext2 /tmp/plik; mount -o loop /tmp/plik /mnt`), a także możliwości tworzenia takich plików z całym układem partycji w środku:

```
losetup /dev/loop1 /mnt/dom0.img
fdisk /dev/loop1
losetup -o 32256 /dev/loop2 /dev/loop1
mkfs.xfs /dev/loop2
mount /dev/loop2 /mnt/1
# ...
umount /mnt/1/
losetup -d /dev/loop2
losetup -d /dev/loop1
```

Problem z while(1) { fork(); }

Jak łatwo się przekonać wykonanie programu zawierającego powyższy kod kompletnie zablokuje system (w skutek zapełnienia pamięci / tablicy procesów). Aby to uniemożliwić należy w pliku /etc/security/limits.conf dodać linijkę `* hard  nproc 200` i w konfiguracji PAM'a (np. /etc/pam.d/common-session) dodać: `session required pam_limits.so`. W tym celu możnaby także użyć starszego pliku /etc/limits (będącego plikiem konfiguracji pakietu shadow), jednak beardziej preferowane jest korzystanie z konfiguracji PAM (podobnie jak z większością opcji z /etc/login.defs).

Zobacz w Sieci: Fork bomb[1790], Cgroups[1791]

Problemy wydajnościowe

Jeżeli podejrzewamy że mamy doczynienia z jakimś problemem typu wydajnościowego to linux oferuje bogaty zbiór narzędzi umożliwiających przyjrzenie się temu bliżej. Polecam zapoznać się z zbiorem informacji na ich temat - Linux Performance[1792].

Wirtualizacja - XEN

Xen[1793] jest monitorem maszyn wirtualnych. Na komputerze na którym funkcjonuje uruchamiany jest najpierw sam Xen, a dopiero potem on uruchamia wskazane jądro systemu operacyjnego (odbywa się to poprzez odpowiedni wpis w bootloaderze, w którym ścieżka obrazu jądra i jego ram dysku są przekazywane jako parametry do ładowanego jądra xen'a). Poniżej zamieszam przykład takiego wpisu wraz z opcjami wystawiającymi konsolę XEN i dom0 na port szeregowy (ttyS1):

```
title           Xen 3.2-1-amd64 / Debian GNU/Linux, kernel 2.6.26-2-xen-amd64
root            (hd0,0)
kernel          /boot/xen-3.2-1-amd64.gz com1=9600,8n1 console=com1,vga dom0_mem=256M
```

```
module          /boot/vmlinuz-2.6.26-2-xen-amd64 root=/dev/sdb1 ro console=tty0
 ↪  console=hvc0
module          /boot/initrd.img-2.6.26-2-xen-amd64
```

Podstawowy system operacyjny uruchamiany przez Xen określany jest jako dom0, inne jako domU. Wspomniany dom0 charakteryzuje się posiadaniem dostępu do całego sprzętu, który nie został jawnie przekazany któremuś z domU (tak samo z pamięcią i czasem procesora). Poszczególne domU uruchamiane (i kontrolowane/zarządzane) są z dom0 przy pomocy komendy `xm`, do startu mogą wykorzystywać nawet ten sam obraz jądra, initrd i (ew.) dysku co dom0 (w tym wypadku w trybie ro lub po prostu kopię). Zobacz `man  xm` i `man  xmdomain.cfg`.

Xen zapewnia komunikację sieciową z systemami wirtualnymi. Realizowane jest to poprzez tworzenie w dom0 wirtualnych interfejsów związanych z interfejsami w domU, interfejsy te noszą oznaczenia `vifX.Y`, gdzie X identyfikuje domU do którego prowadzi dany interfejs, a Y jest kolejnym numerem takiego interfejsu (domyślnie ich liczba dla każdego domU odpowiada liczbie fizycznych interfejsów w dom0). Ich wykorzystanie zależy od wartości `network-script` oraz `vif-script` w `/etc/xen/xend-config.sxp` (pierwsza wskazuje nazwę skryptu odpalanego przy starcie dom0 celem przygotowania go do dostarczania sieci domU, druga wskazuje nazwę skryptu odpalanego przy uruchamianiu domU celem udostępnienia mu sieci, skrypty te znajdują się w `/etc/xen/scripts`). W dalszej części będę zakładał że sieć udostępniana jest poprzez bridge przy ustawieniach: `(vif-script vif-bridge)` i `(network-script network-dummy)` oraz następującej zawartości `/etc/xen/scripts/network-dummy`:

```
/etc/xen/scripts/network-bridge $1 vifnum=0 netdev=eth0
/etc/xen/scripts/network-bridge $1 vifnum=1 netdev=eth1
```

O ilości interfejsów w poszczególnych domU decyduje wpis postaci `vif = [ 'bridge=eth0,mac=00:16:3E:00:00:00', 'bridge=eth1,mac=00:16:3E:00:00:01' ]` w konfiguracji danego domU. W tym przypadku utworzone będą dwa interfejsy o podanych adresach MAC, a odpowiadające im interfejsy `vifX.Y` w dom0 zostaną włączone do wskazanych bridge'ów. Bridge te (eth0 i eth1) zostaną utworzone przez podany powyżej skrypt `network-dummy` i zostaną włączone do nich odpowiednie interfejsy `peth`. Jednak aby `network-bridge` zadziałał poprawnie muszą być przed jego wywołaniem przez xend skonfigurowane poprzez ifup (`/etc/network/interfaces`) odpowiednie interfejsy eth (zamiany nazwy z eth na peth dokonuje wspomniany skrypt przed utworzeniem odpowiedniego bridge'a i przenosi konfigurację z fizycznej karty na bridge'a, a fizyczna karta staje się portem bridge'a).

W wpisach `vif = ...` nie ma formalnego wymogu podawania MAC adresów tworzonych w domU interfejsów, jednak warto to robić aby wiedzieć co jest co oraz aby udev w domU nie przenumerowywał kart (gdyż jeżeli go nie podamy za każdym startem domU będzie generowany losowy z prefiksem 00:16:3E). Warto także zadbać aby udev przypisywał do tych MAC'ow odpowiednie numery interfejsów - eth0 i eth1 (a nie jakieś z kosmosu). Alternatywnymi metodami do opisywanego tutaj bridge'owania jest realizacja zwykłego routingu lub routintgu z NAT pomiędzy siecią dochodzącą do dom0 a siecią domU.

W niektórych wersjach Xen'a występuje problem z wbudowaną synchronizacją zegarów domU z dom0. Jego rozwiązaniem jest rezygnacja z tego mechanizmu (bez niej nie da się ustawić zegara w domU) poprzez wpisanie 1 do `/proc/sys/xen/independent_wallclock` oraz wykorzystanie np. ntpdate do synchronizacji czasu.

W przypadku Xen 3.2 gdy Xen odpalany jest z przekierowaniem swojej konsoli na ttyS* nie należy uruchamiać konsoli jądra ani getty dom0 na tym urządzeniu. W opcjach jądra dla dom0 (tak jak wyżej pokazano) podajemy `console=hvc0`, natomiast getty uruchamiamy na `/dev/hvc0`, ewentualnie na `/dev/console` (w obu wypadkach konfiguracja jak dla tty0 a nie ttyS0). Podobnie należy czynić z getty DomU aby mieć konsolę pod `xm console` (nie odpalamy na /dev/tty1 tylko na /dev/console).

Oprócz xen'a mamy także inne systemy wirtualizacji, takie jak (więcej na stronie z technicznym porównaniem systemów wirtualizacji[1794] oraz w Debian Wiki[1795]):

- (k)qemu[1796] - w zasadzie to jest emulacja platformy x86
- Kernel-based Virtual Machine[1797] (KVM) - najbardziej zbliżony do Xen'a z pośród opisywanych, główną różnicą jest to iż (jak sama nazwa wskazuje) oparty jest na jądrze linux'owym a nie niezależnym "hypervisor"
- VirtualBox[1798] - wirtualizacja na poziomie systemu
- Linux-VServer[1799], OpenVZ[1800], Linux Containers (LXC), User Mode Linux (UML), Schroot - wzbogacone systemy typu chroot/jail, to nie jest prawdziwa wirtualizacja (OpenVZ zawiera dodatkowo zwirtualizowany stos sieciowy), wydają się dobre jako zastosowania "serwer dla każdego użytkownika", zobacz w Sieci: schroot - chroot for any users[1801]

Zobacz w Sieci: Xen - Debian Wiki[1802], Xen Virtualization Essentials[1803].

Nie tylko Debian

Poniżej przedstawię kilka zagadnień administracyjnych specyficznych dla niedebianopodobnych dystrybucji (używaną dystrybucję możemy sprawdzić zaglądając do `/etc/issue`).

W Fedora Core konfiguracja sieci znajduje się w plikach `/etc/sysconfig/network-scripts/ifcfg-INTERFEJS` (dane konfiguracyjne interfejsu) i `/etc/sysconfig/network` (nazwa hosta, włączenie sieci oraz bramka domyślna). pliki te mają składnie bashową (deklarowanie zmiennych) i wykorzystywane są przez skrypt `/etc/init.d/network` (w razie problemów z startem sieci warto zwrócić uwagę czy jest on uruchamiany w aktualnym runlevelu (domyślny to 3). Usługi oprócz skryptów w `/etc/init.d/` mogą być kontrolowane przez `service uslugakomenda`. Oprogramowanie możemy instalować poprzez `yum` (wskazany pakiet z rozwijaniem zależności) lub `rpm` (wskazany plik, może być wskazany URL np. na HTTP). Strefę czasową możemy ustawić np. przy pomocy cp - `cp /usr/share/zoneinfo/Europe/Warsaw /etc/localtime`.

W SUSE do zarządzania systemem, a przede wszystkim instalowania oprogramowania służy `yast` i `yast2`.

Dość interesującym systemem jest PLD. Elementami go wyróżniającymi jest przedewszystkim bardzo wygodny menager pakietów - podek oraz ich znaczne rozdrobnienie. Najistotniejsza konfiguracja znajduje się w plikach `/etc/sysconfig/interfaces`, `/etc/sysconfig/network` i pozostałe `/etc/sysconfig/*` a także `/etc/sysctrl.conf`. System (w odróżnieniu np. od Ubuntu korzystającego z event-based init daemon[1804]) posiada standardowy init (`/etc/inittab`, `/etc/rc*`, `/etc/init.d/*`).

Praktycznie każdy linux umożliwia obok standardowych systemów plików[1805] - ext2[1806]/ext3[1807] używać systemów takich jak XFS[1808] czy ReiserFS[1809], a nawet tak specyficzne rozwiązania jak klastrowy system plików Lustre[1810].

Poradnik linuxowy - porady konfiguracyjne Debiana

Instalacja przez debootstrap ...

Debian umożliwia także instalację systemu z działającego systemu (bez instalatora) - służy do tego celu program debootstrap (aby go zainstalować wydajemy polecenie `apt-get install debootstrap`), który instaluje w wskazanym katalogu (może to być katalog w którym podmontowany jest system plików z innego dysku) podstawowe pakiety dystrybucji (`debootstrap wersja_dystrybucjikatalog_instalacji`). Po wykonaniu tego procesu należy wykonać `chroot katalog_instalacji` (spowoduje to zmianę katalogu głównego dla wszystkich procesów potomnych na wskazany oraz odpalenie /bin/sh pochodzącego z nowego drzewa katalogowego) oraz (już w chrootowanym środowisku) `mount proc /proc -t proc`. Następnie (w przypadku instalacji sieciowej) powinniśmy skonfigurować sieć wraz z DNS (jak to zrobić napisałem powyżej) i przystąpić do instalacji potrzebnych nam pakietów (w szczególności jądra i bootloadera). Po instalacji zalecam także wykonanie `dpkg-reconfigure locales` celem przebudowania informacji o lokalizacji systemu.

Instalacja debootstrap z wykorzystaniem MEPIS 7.0:

```
kernel /boot/vmlinuz aufs init=/bin/bash
  /etc/init.d/udev start
  dhclient eth0
    # lub:
    # ifconfig eth0 up 192.168.24.5 netmask 255.255.0.0
    # ip route add default via 192.168.0.1
    # echo "nameserver 192.168.0.1" > /etc/resolv.conf
  apt-get update
  . /etc/profile
  apt-get install debootstrap
  fdisk ... ; mkfs ... ; mount ... ; debootstrap ...
```

Możliwe problemy

Niekiedy przy instalacji debootstrapem lub w inny sposób może coś nie do końca wyjść. Możemy potem mieć doczynienia z komunikatami typu:

- "Warning: Fake start-stop-daemon called, doing nothing" - rozwiązaniem jest `apt-get install dpkg --reinstall`
- "invoke-rc.d: policy-rc.d denied execution of reload." - rozwiązaniem jest `rm /usr/sbin/policy-rc.d`

Zobacz w Sieci: Własna minidystrybucja[1811], Fully Automatic Installation[1812], linuxcreate[1813].

LiveUSB

Live USB możmy bardzo prosto utworzyć instalując po prostu system z użyciem debootastrapa i uzupełniając go w chroocie o jądro, gruba i potrzebne nam narzędzia.

przygotowanie tablicy partycji

Zasadniczo wystarczył by zwykły MBR z partycją EFI, ale w przykładzie robiona jest hybryda MBR i GPT:

```
# tworzymy tablicę partycji typu gpt
parted $DEV "mklabel gpt"
# tworzy partycję która posłuży do wgrania gruba (musi mieć co najmniej 128 kiB)
# należy zaznaczyć iż nie jest to partycja /boot
# jest to surowe (bez filesystemu) miejsce na dysku gdzie zostanie wgrany fragment gruba
↪ normalnie wgrywany do MBR
parted $DEV "mkpart grub 0 2MB";
# ustawiamy dla tej partycji flagę "GRUB BIOS partition"
parted $DEV "set 1 bios_grub on"
parted $DEV "mkpart efi 2MB 200MB";
set 2 boot on
# resztę dysku możemy podzielić wg uznania w tym przykładzie robimy jedną dużą partycję o
↪ nazwie rootfs
parted $DEV "mkpart rootfs 200MB 100%";
# przy pomocy gpart utworzenie hybrid MBR
gpart $DEV
# r ->
#      o
#      h ->
#           1 2 3
#           n 83 n
#           ef y
#           83 n
#      o
#      w ->
#           y
mkfs.vfat -F 32 ${DEV}2
mkfs.xfs ${DEV}3
```

Instalacja systemu

```
# zamontowanie systemu plików z USB i debootstrap
mount ${DEV}3 ${MNT}
debootstrap stable ${MNT} ftp://ftp.icm.edu.pl/pub/Linux/debian/
# zamontowanie /proc, /sys i /dev na potrzeby chroot'a
mount -t proc  proc  $MNT/proc
mount -t sysfs sysfs $MNT/sys
mount -o rbind /dev  $MNT/dev
# chroot
chroot $MNT
```

Uwaga: wszystkie następne kroki wykonywane są w chroot

podstawowa konfiguracja APT

```
cat << EOF > /etc/apt/apt.conf.d/03misc
# nie trzyma polecanych i sugerowanych jak zalezności
APT::AutoRemove::RecommendsImportant "false";
APT::AutoRemove::SuggestsImportant "false";
# nie instaluje automatycznie olecanych i sugerowanych
APT::Install-Recommends "false";
APT::Install-Suggests "false";
# wyłączenie wyszukiwania przyrostowego w aptitude
aptitude::UI::Incremental-Search "false";
# wyłączenie pobierania aktualizacji listy pakietów jako pdiff
# (polecam jeżeli stosunkowo rzadko aktualizujemy listy pakietów np. na testingu)
Acquire::PDiffs "false";
EOF
apt-get install aptitude
```

Grub i jądro

```
# instalacja pakietów gruba, jądra oraz narzędzi efi
aptitude install grub2 grub-efi-amd64-bin efivar efibootmgr linux-image-amd64
# instalacja GRUBa w MBR
grub-install --root-directory=/boot/   --target=i386-pc    $DEV
# instalacja GRUBa dla UEFI
mkdir /boot/efi
mount ${DEV}2 /boot/efi
grub-install --efi-directory=/boot/efi --target=x86_64-efi
mkdir /boot/efi/EFI/BOOT
cp /boot/efi/EFI/debian/grubx64.efi /boot/efi/EFI/BOOT/bootx64.efi
# plik konfiguracyjny GRUBa
UUID=`grub-probe -d ${MNT} -t fs_uuid`
cat << EOF > /boot/grub/grub.cfg
# korzystanie z konsoli na porcie szeregowym
#serial --speed=115200 --unit=1 --word=8 --parity=no --stop=1
#terminal_input serial console
#terminal_output serial console
# za to na którym RS jest grub odpowiada --unit w serial
#
# korespondujące opcje uruchamiania dla Linuxa:
#   module    /boot/vmlinuz [...] console=tty0 console=ttyS1,115200n8
# korespondujące opcje uruchamiania dla XENa:
#   multiboot /boot/xen.gz  [...] com2=115200,8n1 vga=mode-0x0319 console=com2,vga
#   module    /boot/vmlinuz [...] console=tty0 console=hvc0
#
# w opcjach kernela szczególną rolę odgrywa ostatni podany terminal (ostatnia opcja
↪  console)
# jest on związany z /dev/console i na nim pokazywany jest init, ponadto należy
↪  uruchamiać getty
# na odpowiednim urządzeniu poprzez wpis w inittab np.:
#   T0:23:respawn:/sbin/getty -L /dev/ttyS1 115200 vt100
# lub (dla wariantu XENowego, UWAGA: w przykładzie tylko runlevel 4 i 5):
#   T1:45:respawn:/sbin/getty 38400 hvc0
# timeout w sekundach dla domyślnej pozycji menu
set timeout=60
set default="0"
# ładowanie modułów - typ tablicy partycji
insmod part_gpt
insmod part_msdos
# ładowanie modułów - RAID
insmod mdraid1x # mdraid metadata 1.x
```

```
insmod mdraid09 # mdraid metadata 0.9
#insmod raid5rec # faulty RAID4/5
#insmod raid6rec # faulty RAID6
# ładowanie modułów - LVM
insmod lvm
# ładowanie modułów - File System
insmod xfs
# ustawienie root'a
search --no-floppy --fs-uuid --set=root $UUID
# ustawienia wizualne
if [ x\$feature_default_font_path = xy ] ; then
    font=unicode
else
    font="/usr/share/grub/unicode.pf2"
fi
if loadfont \$font ; then
  if [ x\$feature_all_video_module = xy ]; then
    insmod all_video
  else
    insmod efi_gop
    insmod efi_uga
    insmod ieee1275_fb
    insmod vbe
    insmod vga
    insmod video_bochs
    insmod video_cirrus
  fi
  insmod gfxterm
  insmod gettext
fi
set menu_color_normal=white/red
set menu_color_highlight=yellow/black
set pager=1
terminal_output gfxterm
# pozycje menu
menuentry 'Debian GNU/Linux USB-LIVE by rrp@opcode.eu.org' {
  echo    'Loading Linux 3.16.0-4-amd64 ...'
  linux   /boot/vmlinuz-3.16.0-4-amd64 root=UUID=$UUID rootdelay=5 ro
  echo    'Loading initial ramdisk ...'
  initrd  /boot/initrd.img-3.16.0-4-amd64
}
menuentry 'Debian GNU/Linux USB-LIVE by rrp@opcode.eu.org init=/bin/bash' {
  echo    'Loading Linux 3.16.0-4-amd64 ...'
  linux   /boot/vmlinuz-3.16.0-4-amd64 root=UUID=$UUID rootdelay=5 ro init=/bin/bash
  echo    'Loading initial ramdisk ...'
  initrd  /boot/initrd.img-3.16.0-4-amd64
}
EOF
```

Instalacja przydatnych narzędzi i programów

```
SYSBASE=" tmux vim-nox less apcalc bc python gawk gpm busybox
          coreutils bsdutils bsdmainutils diffutils findutils grep sed tar util-linux
 ↪  gzip
          bash-completion manpages file konwert whiptail bzip2 p7zip p7zip-full
 ↪  p7zip-rar"
SYSTOOL=" lsof procps psmisc sudo locales sysstat iotop itop htop debootstrap picocom
 ↪  cutecom"
HWTOOL="  parted gdisk gnu-fdisk mdadm lvm2 dmsetup multipath-tools xfsprogs"
```

```
HWDIAG="  pciutils usbutils lsscsi dmidecode sdparm smartmontools hddtemp hdparm ipmitool
 ↪  lm-sensors pcmciautils"
NETBASE=" isc-dhcp-client resolvconf net-tools bridge-utils vlan iproute ethtool
 ↪  iputils-ping iputils-tracepath
          tcptraceroute mtr-tiny telnet netcat netcat6 netcat-traditional socat picocom
 ↪  nmap tcpflow"
NETTTOL=" wget curl w3m w3m-img links2 rsync ntpdate bind9-host ssh sshfs"
NETDIAG=" arping arp-scan ndisc6 ipv6toolkit inetutils-traceroute paris-traceroute
          netdiag vnstat iftop sipcalc tcpdump ngrep dnstop dnstracer dnsutils dnswalk"
NETWIFI=" wireless-tools wpasupplicant rfkill"
XTOOL="   xorg openbox iceweasel xpdf"
aptitude install $SYSBASE $SYSTOOL $HWTOOL $HWDIAG $NETBASE $NETTTOL $NETDIAG $NETWIFI
 ↪  $XTOOL
aptitude clean
```

Konfiguracja

```
# w /etc/fstab ustawiamy rootfs na read-only, a /tmp na tmpfs
# /var/run i /var/lock są współcześnie linkami do /run na tmpfs
cat << EOF > /etc/fstab
/dev/disk/by-uuid/$UUID  /     xfs    ro        0  0
tmpfs                    /tmp  tmpfs  defaults  0  0
EOF
# linkujemy /proc/mounts do /etc/mtab
ln -s /proc/mounts /etc/mtab
# automatyczne podnoszenie sieci
echo "allow-hotplug eth0" > /mnt/1/etc/network/interfaces.d/eth0
echo "iface eth0 inet dhcp" > /mnt/1/etc/network/interfaces.d/eth0
# ssh pozwala na logowanie na root'a i nie jest automatycznie uruchamine
sed -e 's|^[# ]*PermitRootLogin.*$|PermitRootLogin yes|' -i /etc/ssh/sshd_config
systemctl  disable  ssh.service
# konfiguracja do rootfs w ro itp
systemctl  mask  hwclock-save.service systemd-random-seed.service
 ↪  systemd-update-utmp-runlevel.service systemd-update-utmp.service udev-finish.service
ln -s /tmp/Xorg.0.log /var/log/Xorg.0.log
rm /usr/lib/modules-load.d/ipmievd.conf
# konfiguracja użytkownika
cat << EOF > /root/prepareChroot.sh
#!/bin/bash
if [ \$# -ne 2 ]; then
        echo USAGE: \$0 device mountpoint
        exit
fi
DEV=\$1
MNT=\$2
mount \$DEV \$MNT
mount -t proc  proc  \$MNT/proc
mount -t sysfs sysfs \$MNT/sys
mount -o rbind /dev  \$MNT/dev
chroot \$MNT
EOF
chmod +x /root/prepareChroot.sh
cat << EOF > /root/.bash_history
mount -o remount,rw /
systemctl start ssh.service
/root/prepareChroot.sh /dev/sdb1 /mnt/1
chroot /mnt/1
EOF
ln -s /root/.bash_history /.bash_history
```

Modyfikowanie / przebudowanie pakietu:

Aby zbudować jakis pakiet ("pakiet_do_poprawki") należy pobrać jego wersję źródłową (`apt-get source pam`), nanieść poprawki (albo na kod źródłowy albo tylko na opcje budowania umieszczone w "rules") a następnie zbudować pakiet binarny (`dpkg-buildpackage -rfakeroot`) i zainstalować go lub wypakować z niego potrzebne pliki. W przypadku pakietów budowanych na nasze potrzeby pomocny może być też program "debchange", umożliwiający automatyzację zmian w changelog - np. `debchange -v "1.0."`date +"%Y%m%dT%H%M%S"` "OPIS" && dpkg-buildpackage -rfakeroot`

Budowanie modułów dla swojego jądra

Często nie ma paczek z zbudowanymi modułami dla używanej przez nas wersji jądra, niekiedy takich modułów nie ma w ogóle. Należy wtedy zainstalować pakiet XXXX-source (np. shfs-source, o którym poniżej), w katalogu `/usr/src/` pojawi się plik z spakowanymi źródłami danego modułu. Należy po przejściu do wspomnianego katalogu rozpakować źródła (najczęściej: `tar -xjf xxxx.tar.bz2 lub tar -xzf xxxx.tar.gz`). Następnie przejść do katalogu z nagłówkami odpowiedniego jądra (musi być zainstalowany odpowiedni pakiet linux-headers-YYYYZZZZ, gdzie YYYY wersja jadra, a ZZZZ to określenie wersji pakietu jądra - np: `-1-686-smp`) - `/usr/src/linux-headers-YYYYZZZZ`. I tam wykonać: `make-kpkg --rootcmd=fakeroot --append-to-version=ZZZZ modules-image`. Po wykonaniu tego polecenia wracamy do `/usr/src/` i tam czekają na nas gotowe do zainstalowania (np. przez dpkg -i) pakiety Debiana (*.deb).

Budowanie (zmodyfikowanego) jądra

Niekiedy zachodzi potrzeba zbudowania zmodyfikowanego (pod względem konfigu lub z nałożonymi łatami) jądra. Poniżej przedstawię drogę do zbudowania jądra 2.6.24.4 z nałożonymi łatami IMQ oraz wybranymi z łat patch-o-matic. Jądro będzie budowane metodą Debiana, a poniższy opis należy traktować jako przykład który należy dostosować do własnych potrzeb.

```
####################
###  POBIERANIE  ###
####################
export KERNEL_DIR="$PWD/2.6.24"
export IPTABLES_DIR="$PWD/iptables-1.4.0"
wget http://www.linuximq.net/patchs/iptables-1.4.0-imq.diff
mv iptables-1.4.0-imq.diff iptables-imq.diff
wget http://www.linuximq.net/patchs/linux-2.6.24-imq.diff
mv linux-2.6.24-imq.diff linux-imq.diff
wget http://ftp.netfilter.org/pub/patch-o-matic-ng/snapshot/patch-o-matic-ng-20080405.
 ↪  tar.bz2
tar -xjf patch-o-matic-ng-20080405.tar.bz2
mv patch-o-matic-ng-20080405 patch-o-matic-ng
apt-get source iptables
wget ftp://ftp.icm.edu.pl/pub/Linux/kernel/v2.6/linux-2.6.24.4.tar.gz
tar -xzf linux-2.6.24.4.tar.gz
mv linux-2.6.24.4 2.6.24
cp /boot/config-2.6.24-1-686 $KERNEL_DIR/.config
#############
###  IMQ  ###
#############
patch -p0 < linux-imq.diff
patch -p0 < iptables-imq.diff
######################
###  patch-o-matic  ###
######################
pushd patch-o-matic-ng;
./runme pending
## base ##
# dopasowywanie u32; od 2.6.23 w jajku
```

```
#./runme u32
# grupy adresów obsługiwane programem ipset
./runme set
## extra ##
# ustawianie trasy routibgu z poziomu iptables
./runme ROUTE
## external ##
./runme --download
# masowe markowanie
./runme IPMARK
# reguły filtrowania true/false sterowanego przec /proc
./runme condition
# limity połączeń; od 2.6.23 w jajku
#./runme connlimit
# filtrowanie po czasie
./runme time
popd
############################
###  budowanie iptables  ###
############################
pushd $IPTABLES_DIR
chmod +x $IPTABLES_DIR/extensions/.IMQ-test*
debchange -l "imq" "IMQ"
dpkg-buildpackage -rfakeroot
popd
#########################
###  budowanie jądra  ###
#########################
pushd $KERNEL_DIR
make oldconfig;
vi .config
make-kpkg oldconfig
make-kpkg clean
fakeroot make-kpkg --revision imq1 --initrd kernel_image
```

Altenatyną drogą (bardziej szczegułowo opisywaną na debianowym wiki[1814]) byłoby:

```
apt-get source linux-2.6_3.2.6-1
dpkg-source -x linux-2.6_3.2.6-1.dsc
cd linux-2.6-3.2.6
fakeroot make -f debian/rules.gen setup_amd64_none
# pachte zmiany konfigu (w debian/build/...)
# ewntualne zmiany w makefile (cel binary-arch_amd64_none) aby budować tylko tą paczkę co
↪  potrzebujemy
fakeroot make -f debian/rules.gen binary-arch_amd64_none
# jeżeli chemy tylko debugować jakiś moduł to możemy komędę tę przerwać w okolicy:
#  env -u ABINAME -u ABINAME_PART -u ARCH -u FEATURESET -u FLAVOUR -u VERSION -u
↪  LOCALVERSION DISTRIBUTION_OFFICIAL_BUILD=1 DISTRIBUTION_UPLOADER=ben@decadent.org.uk
↪  DISTRIBUTION_VERSION="3.2.6-1" make -C 'debian/build/build_amd64_none_amd64
# wykonać ta komendę manualnie i budować moduły z użyciem normalnego make
↪  /katalog/modul.ko
```

Konfiguracja wstępna Debiana

Na świeżo zainstalowanym systemi Debiana warto dodać pakiety: gawk, less, psmisc, vim, vim-scripts, bc, iproute, openssh-client, openssh-server. Usunąć można natomiast vim-tiny i mawk Konieczne może się także okazać ustawienie właściwej strefy czasowej (konfiguracja pakietu tzdata), locale itp. Jeżeli zamierzamy korzystać z LVM oraz RAID potrzebne są także pakiety: mdadm, lvm2.

Osobiście polecam np. następujący zestaw komend: `aptitude --without-recommends install atop bind9-host bridge-utils busybox bzip2 dnsutils ebtables file gawk grub hdparm htop`

```
iftop iotop ipmitool iputils-tracepath itop less mailx netcat netcat6 nfs-kernel-server
ntpdate nullmailer openssh-client openssh-server openssl paris-traceroute perl psmisc
psutils python rssh rsync screen sdparm sudo sysstat tcptraceroute traceroute-nanog
udev unrar-free unzip vim vnstat w3m locales ethtool linux-image-2.6-XXXXX dmsetup
debootstrap ebtables hddtemp lm-sensors lvm2 mdadm openipmi smartmontools arptables &&
dpkg-reconfigure locales
```

Poradnik linuxowy - porady konfiguracyjne Desktop

Ustawienia serwera X'ów - ładniejsze czcionki, wiele monitorów, ...

Jeżeli nasze czcionki, pomimo przyzwoitej rozdzielczości wyświetlanego obrazu, nie wyglądają zbyt ładnie, są zbyt małe, opcje antyaliasingu często wręcz pogarszają sytuację, przyczyną może być zbyt mała rozdzielczość optyczna wyświetlanych obrazów (ta wyrażana w ppi[1815], niekiedy niepoprawnie nazywanym niekiedy dpi[1816]). Rozdzielczość tą możemy sprawdzić poleceniem `xdpyinfo | grep resolution`, rozdzielczość ta wraz z rozdzielczością całego ekranu przekłada się na jego (wirtualne) wymiary (które możemy sprawdzić `xdpyinfo | grep dimensions`). Aby temu zaradzić w sekcji Monitor pliku konfiguracyjnego serwera X należy określić właśnie tą wielkość ekranu (w mm): `DisplaySize X Y` gdzie X i Y to odpowiednie wymiary obliczone z zależności X,Y = pikseleX,Y / rozdzielczosc_ppi * 25.4 i przybliżone w dół do liczby całkowitej. Dobrą wartości ppi wydaje się być 96.

X serwer umożliwia także konfigurację środowiska wielomonitorowego. Wymaga to oczywiście odpowiedniego wsparcia sprzętowego - karta z wieloma wyjściami (której sterownik jest zgodny z xrandr lub xinerama), albo zestaw niekłócących się ze sobą kart. Obecnie metoda konfiguracji typu xinerama (ServerLayout złożony z wielu Screen'ów - każdy na osobnym urządzeniu, w szczególności zwielokrotnieniu Device pojedynczej karty z podaniem różnych wartości Screen oznaczających wyjście danej karty) zaczyna odchodzić do historii na rzecz "półautomatycznej" konfiguracji xrandr. Wielomonitoring poprzez xrandr możemy obsługiwać dynamicznie lub z stałą konfiguracją w xorg.conf, przykład takiej konfiguracji dla karty obsługiwanej przez sterownik radeonhd przedstawiam w xorg.conf[1817].

Komenda xrandr może posłużyć do zmiany rozdzielczości w locie, przełączania ekranów a także ich wyłączania. X serwer umożliwia także uśpienie monitora (przełączenie w tryb oszczędzania energii), możemy tego dokonać komendą: `killall xscreensaver ; sleep 1; xset dpms force off`, w odróżnieniu od wyłączenia z użyciem xrandr monitor włączy się automatycznie po ruszeniu myszką lub naciśnięciu klawiatury. Xset umożliwia także wyłączenie automatycznego włączania wygaszacza ekranu i przechodzenia w tryb uśpienia (przydatne gdy nasz odtwarzacz filmów nie potrafi tego zablokować): `xset s off; xset -dpms`. Niektóre operacje (np. regulacja jasności podświetlenia w laptopie) wymagają bezpośredniego odwołania do podsystemu /sys, np. w przypadku Panasonic CF-18: `echo 255 > /sys/class/backlight/panasonic/brightness` ustawi maksymalną jasność podświetlenia (minimalną zapis zera).

Zdalny pulpit

Oprócz możliwości odpalenia zdalnych programów na lokalnym X-serwerze (np. przez ssh z opcją -X), odpalenia programu na sesji innego użytkownika (gdy zezwolił nam na to poprzez xhosts) czy też podłączenia się do zdalnego serwera X'ów i odpalenia na nim sesji (X-terminale), mamy także możliwość zdalnego podłączenia się do sesji X-serwrera. W tym celu po stronie zdalnej należy uruchomić serwer vnc (w wariancie x11vnc), np. przy pomocy komendy `x11vnc -passwd 123 -forever --display :0`. Po stronie lokalnej potrzebować będziemy klienta vnc - `vncviewer -encodings Hextile IP.lubdomena.strony.zdalnej:5900`. Zobacz też VNC[1818], NX[1819].

Niekiedy wystarczającym rozwiązaniem może okazać się współdzielenie klawiatury i myszy poprzez sieć - można do tego użyć programu synergy[1820].

Konfiguracja i sztuczki z managerem okien

Często pomimo zastosowania całkiem bogatych opcji konfiguracyjnych managera okien nie udaje się uzyskać pożądanego zachowania niektórych okien. Chyba najbardziej konfigurowalnym managerem okien jest kwin (np. w fluxboxie brakowało mi rozbudowanych opcji regulacji wielkości okna, możliwości wywołania programu po wyświetleniu nowego okna, a także zarządzania przyciskami na belce tytułowej dla poszczególnych okien - kwin też niestety tego nie posiada) i dlatego w tych przykładach oprę się na nim. Pomocne mogą się wtedy okazać programy takie jak: `devilspie` czy też `wmctrl` oraz ... system obsługi zdarzeń, który (przynajmniej w kde) nie tylko może odtworzyć dźwięk przy otwarciu okna ale również wykonać jakiś program.

`wmctrl` umożliwia m.in. przeniesienie wybranego okna na bieżący pulpit i aktywację go, ukrywanie i pokazywanie okien, ustawianie pomijania paska zadań, przypinanie okien. Na przykład poniższy kod spowoduje przeniesienie wszystkich okien Iceweasel na pulpit 4:

```
wmctrl -l | grep Iceweasel | while read id reszta; do wmctrl -i -r $id -t4; done
```

W skrypcie[1821], który wywoływany jest w zdarzeniu[1822] związanym z wyświetleniem nowego okna, jest on uruchamiany gdy wywołane było okno programu Psi i aktywuje je przełączając się na jego pulpit. Konieczność stosowania takiej sztuczki wynika z tego iż przy normalnej konfiguracji[1823] kwin'a, gdy wymuszamy otwarcie na wskazanym pulpicie to przy wywoływaniu z tray'a nie następuje przełączenie na niego i niekiedy prowadzi to nawet do zgubienia tego okna. Trudniejsze do uzyskania (ale myślę że także możliwe) byłoby wymuszenie na kmail'u aby po wywołaniu z tray'a otwierał się na bieżącym pulpicie (czego w odróżnieniu od standardowego zachowania psi nie che normalnie robić; co najciekawsze np. w fluxboxie zachowanie jest dokładnie odwrotne).

Przy grzebaniu w menagerze okien (czy wogóle środowisku graficznym) warto popatrzeć w pliki `~/.profile`, `~/.xprofile`, `.Xmodmap`, `.Xkbma`, `.Xresources`, `.xsession`, `.xsessionrc`, `.Xsession` oraz `/etc/X11/Xsession*`

Ekrany dotykowe zgodne z GUNZE

Dawniej w przypadku ekranów dostępnych poprzez port szeregowy (występujących np. w laptopach Panasonic CF-M34) można było zastosować dedykowany sterownik X-serwera [homepage[1824]], jednak sterownik ten nie che współdziałać z nowszymi wersjami Xorg. Dlatego konieczne jest użycie odpowiedniego modułu jądra (używanego również przez ekrany z interfejsem USB - występujących np. w Itronix GoBook IX250). Do skonfigurowania tego modułu potrzebna jest zmodyfikowana wersja programu inputattach (z pakietu joystick[1825]). Modyfikacja polega na dodaniu do `input_types[]` wpisu:

```
{ "--gunze",                "-gun",               B9600,
↪  CS8,                       SERIO_GUNZE,        0,         0,         0,        NULL },
```

Po skompilowaniu i dodaniu do skryptów startowych uruchomienia tego programu: `inputattach -gun /dev/ttyS3` dalsza konfiguracja (identyczna jak dla przypadku ekranów USB) odbywa się w xorg.conf i związana jest głównie z kalibracją urządzenia. Poniżej zamieszczam stosowne fragmenty xorg.conf z komentarzem dotyczącym procesu kalibracji

```
Section "InputDevice"
        Identifier "dummy"
        Driver "void"
        Option "Device" "/dev/input/mice"
EndSection
Section "InputDevice"
        Identifier        "Toutchpad"
        Driver                "mouse"
        Option          "Device" "/dev/input/mouse1"
EndSection
Section "InputDevice"
        Identifier        "Touchscreen"
        Driver          "evtouch"
        Option          "Device"           "/dev/input/event1"
        Option          "ReportingMode"    "Raw"
        Option          "DeviceName"       "touchscreen"
        Option          "MoveLimit"        "5"
        # kalibracja uzyskana w wyniku dzialania /usr/lib/xf86-input-evtouch/calibrate.sh
        # kalibracje wykonujemy z aktywana opcja:
```

```
        Option "Calibrate" "1"
        # wg opisu z /usr/share/doc/xserver-xorg-input-evtouch/README.calibration
        # wynik kalibracji zapisywany jest w $PWD/out.txt umieścić poniżej
        #
        # aby ev_calibrate dzialal do czasu rozwiazania Debian Bug #482195 nalezy
↪  wykonac:
        # cp /usr/share/xf86-input-evtouch/empty_cursor.xbm /
EndSection
Section "ServerLayout"
        Identifier      "Default Layout"
        Screen          "Default Screen"
        InputDevice     "dummy"
        InputDevice     "Keyboard"
        InputDevice     "Toutchpad"
        InputDevice     "Touchscreen"
EndSection
```

Czytniki kart pamięci (USB Mass Storage)

Urządzenia te obsługiwane są bez większych problemów, emulują one urządzenia SCSI. Warto jednak przy podłączaniu takiego czytnika (najlepiej od razu z włożoną kartą) przełączyć się na konsolę i zobaczyć co jest tam wypisane (m.in. jako jakie urządzenie widoczna jest karta). W przypadku problemów z ustaleniem typu partycji warto skorzystać z fdiska, najczęściej `fdisk /dev/sdX` (X = a,b,...) i wyświetlić listę partycji (klawisz p). Warto też umieścić odpowiednie wpisy w /etc/fstab np:

```
/dev/sdd1      /media/mmc      vfat    rw,user,noauto      0    0
```

Nagrywanie CD/DVD i zapis pakietowy UTF

Współcześnie (w odróżnieniu od stanu z przed kilku lat) programy nagrywające CD/DVD nie wymagają emulacji SCSI gdy wykorzystywany jest sprzęt IDE. Ponadto warto zainteresować się także zapisem pakietowym (UDF[1826]). Korzystanie z niego wygląda następująco:

```
# przygotowanie płytki
cdrwtool -d /dev/cdrom -q
pktsetup cdrom /dev/cdrom
mkudffs --utf8 /dev/pktcdvd/cdrom
# montowanie płytki
pktsetup cdrom /dev/cdrom
mount -t udf -o rw,noatime /dev/pktcdvd/cdrom /cdrom
chmod o+w /cdrom
#teraz możemy pisać i czytać
# odmontowywanie płytki
umount /cdrom
pktsetup -d cdrom
eject /dev/cdrom
```

Płytę także można montować normalnie (tylko do odczytu) w standardowy sposób przy czym podać należy jako system plików udf - w `/etc/fstab` można wpisać `udf,iso9660`. Można także bawić się z zapisem gołych danych na płycie (np. pliku tar).

Kasowanie ”niegrzecznych” płytek (takich przy których kasowaniu np. k3b ma problemy) warto spróbować wykonać bezpośrednio z pod konsoli komendą `wodim blank=all`.

Laptopy - pcmcia, itp

Aby możliwe było korzystanie z kart PCMCIA wymagany jest pakiet pcmciautils. Podobne do niego pakiety (pciutils i usbutils) są przydatne ale nie są niezbędne dla poprawnego działania odpowiednich podsystemów.

Dostrajanie KDE

Autostart, menu i skojarzenia plików

Katalogiem z którego programy uruchamiane są automatycznie po starcie KDE jest `$HOME/.kde/Autostart`. Można w nim umieszczać dowiązania symboliczne do programów oraz skrypty uruchamiające programy z zadanymi parametrami (w tym wypadku należy pamiętać o ustawieniu atrybutu wykonywalności oraz rozpoczęciu skryptu linia określającą interpretator np: `#!/bin/bash`).

Z kolei konfiguracja menu przechowywana jest w katalogu `$HOME/config/menus/` (plik `applications-kmenuedit.menu`). Jest ona zapisana w XML[1827]'u jako różnica w stosunku co do konfiguracji domyślnej opartej na kategoriach (patrz `/etc/xdg/menus/`). Natomiast opisy poszczególnych pozycji menu ("skróty do programów") odpowiedzialne za opisy, ikony, oraz skojarzenia z obsługiwanymi typami MIME[1828] przechowywane są w postaci plików .desktop w `$HOME/.local/share/applications/` (całosystemowe: `/usr/share/applications/`) oraz w starym systemie menu (pliki tam umieszczone nie maja wpisu o kategorii - odpowiada jej katalog w którym znajduje się plik) w `$HOME/.kde/share/applnk` i `/usr/share/applnk/`. Podobnie informacje o pozycjach katalogowych znajdują się jako pliki *.directory w `$HOME/.local/share/desktop-directories` i `/usr/share/desktop-directories/` oraz w podkatalogach starego systemu menu. Zobacz w Sieci: opis plików .desktop/.directory[1829].

Pliki .desktop

Już wspomniałem że za skojarzenia plików w KDE odpowiedzialne są te same pliki .desktop co tworzą menu (oczywiście mogą być pozycję menu bez skojarzonych typów plików - brak lub pusty wpis `MimeType`, jak też pozycje nie wyświetlane w menu - katalog .hidden). Skojarzenia możemy edytować z menu edycji skojarzeń plików, z właściwości danego pliku .desktop oraz przez tekstową edycję tych plików. Jeżeli przy dwóch pierwszych z tych metod są problemy z zapamiętaniem należy zmienić też inne pole (np. komentarz) - wtedy zmiany MIME zostaną zapamiętane.

Poprzez operowanie na plikach .desktop możemy też rozszerzać menu "działania" z Konqueror'a (pliki z odpowiednim wpisem w `$HOME/.kde/share/apps/konqueror/servicemenus`; np. umieszczenie tam pdflatex.desktop[1830] umożliwi nam budowanie PDF'ów z plików TeX z pod prawego przycisku myszy.

Warto też zwrócić uwagę na katalog `$HOME/.kde/share/services/`, gdzie przechowywane są ustawienia skojarzeń dla opcji "podgląd" w Konquerorze oraz jego podkatalog `searchproviders` gdzie przechowywane są informacje o konfigurowanych przez użytkownika skrótach wyszukiwania (wpisywanych w pasku adresu tak jak nazwy protokołów takich jak: http: ftp: mailto: man: info: audiocd: (rippowanie CD) tar: zip: imap: (dostęp do konta IMAP) pop3: smb: (przeglądanie sieci Windows) programs: (przeglądanie menu KDE) czy już zdefiniowanych wyszukiwań jak gg: (Google) wp: (Wikipedia)). Zamieszczam także kilka plików *.desktop oraz jeden skrypt, sprawiające wygodniejszą obsługę archiwów: archiwa.tgz[1831] (skrypt należy umieścić w $HOME/.config/ a dołączone plik .desktop w $HOME/.kde/share/applnk/.hidden/). Z kolei w menu.desktop[1832] pokazuje jak można utworzyć menu które znajdzie się jako podmenu w menu kontekstowym akcji lub na poziomie równym z nim.

compres_tmp.sh

```
#!/bin/bash

# skrypt ten należy umieścić w $HOME/.config/ a dołączone plik .desktop w $HOME/.⤸
kde/share/applnk/.hidden/

# wariant z pakowniem i ladna nazwa
name=$(mktemp -d)
file_name=$(basename $2 | awk -F. '{NF=NF-1; OFS="."; print $0}')
cd "$name"

$1 -dc "$2" > $name/$file_name
#konqueror –profile webbrowsing "file:$name/$file_name"
kfmclient openProfile webbrowsing "file:$name/$file_name"

zenity --question --title="$2 - zachować zmiany?" --text="Jeżli edytowałeś plik z ⤸
archiwum $2
To po zakończeniu edycji naciśnij:
```

```
* OK aby zachować zmiany w archiwum
* Anuluj aby nie zachować zmian w archiwum

Jeżeli nie modyfikowałeś tego pliku (i nie masz takiego zamiaru) wybierz Anuluj

Po wybraniu Anuluj wszelkie zmiany zostaną utracone !
Wszystkie zmiany zapisane po zamknięciu tego komunikatu zostaną utracone !"

if [ $? = 0 ]; then
 $1 -c "$file_name" > "$2"
fi
cd /tmp
rm -Rf "$name"

# wariant bez pakownia
#name=$(mktemp) && $1 -dc "$2" > $name && konqueror --profile webbrowsing "file:$name" && cd ⤸
/tmp && rm -Rf "$name"
```

Mymenu - czyli generowanie menu KDE oraz menu kontekstowych dla różnych typów plików

Jest to system generacji menu dla KDE mojego autorstwa. Korzysta on z katalogowej struktury w której umieszczane są plik .desktop poszczególnych aplikacji. Każdy z katalogów odpowiada kategorii w menu i posiada plik .directory który opisuje wygląd kategorii (ikonę, nazwę itp). Nazwy plików i katalogów nie mają większego znaczenia, ale warto je rozpoczynać od liczby celem zapewnienia odpowiedniego sortowania w menu (odbywa się ono właśnie w oparciu nie o nazwy kategorii/aplikacji a plików). System generuje także pliki .desktop menu akcji Konquerora. Konfiguracja przypisań aplikacji do menu kontekstowego zależnego od typów mime odbywa się w `service_menus.cfg`. Jeżeli w tym pliku podano cały katalog wypełniane ono jest w kolejności określonej poprzez nazwy plików na niego się składających.

Pliki .desktop mogą posiadać określenia typów mime które obsługują i będą one wykorzystywane do generacji klasycznego systemu skojarzeń w konquerorze (niestety mamy tutaj tylko ograniczony wpływ na kolejność - poprzez parametr `InitialPreference=`. Aby nie dublować wpisów w menu "otwórz w" (różne pliki .desktop wskazujące na tą samą aplikację) zaleca się umieszczenie w standardowej lokalizacji plików opisu aplikacji plików .desktop z standardowymi nazwami (aby mogły nadpisywać ogólnosystemowe) ale bez z pustym atrybutem `MimeType=`; możliwe jest też usunięcie plików ogólnosystemowych. W standardowym systemie skojarzeń Konquerora można także wykorzystać mechanizm mailcap (`man mailcap`, `man run-mailcap`, `man update-mime`) poprzez utworzenie wpisów z maksymalnym np. `InitialPreference=9` wskazującym na wrapper(-y) tego systemu.

Skrypt instalujący system (mymenu_install.sh) dodaje dowiązanie do własnego pliku menu w pliku z standardowym menu KDE. Ze względu na czytelność zaleca się przeniesienie oryginalnych kategorii do jakiejś zbiorczej kategorii (przykład[1833]). Całość systemu można pobrać jako archiwum tar.gz[1834].

mymenu_install.sh

```
#!/bin/bash

# funkcja przetwarza katalog, dla podkatalogów wywołuje się rekurencyjnie
# $1 - katalog, $2 - zagnieżadzenie kodu
print_dir() {
 # generowanie zawartości - submenus
 for f in $1/*; do
  if [ -f $f/.directory ]; then
   bn=`basename $f`
   entry=${bn/[0-9]*-/}
   echo "$2<Menu>"
   echo " $2<Name>$entry</Name>"
   echo " $2<Directory>$f/.directory</Directory>"
   echo " $2<AppDir>$f/</AppDir>"
   print_dir $f " $2"
```

```
   echo "$2</Menu>"
  fi
 done;

# generowanie zawartości - menu entry
 echo "$2<Include>"
for f in $1/*.desktop; do
  if [ -f $f ]; then
   bn=`basename $f`
   echo " $2<Filename>$bn</Filename>"
  fi
 done;
 echo "$2</Include>"

# generowanie układu
 echo "$2<Layout>"
pushd $1 > /dev/null
ls | while read f; do
  entry=${f/[0-9]*-/}
  if [ -d $f ]; then
   echo " $2<Menuname>$entry</Menuname>"
  elif [ -f $f ]; then
   if [ "`expr $entry : '.*\(\.desktop\)$'`" != "" ]; then
    echo " $2<Filename>$f</Filename>"
   elif [ "$entry" = "SEPATARTOR" ]; then
    echo " $2<Separator/>"
   fi
  fi
 done
 popd > /dev/null
if [ "$2" = " " ]; then # gdy menu głównego poziomu
  echo -e ' <Separator/>\n <Merge type="menus" />\n <Merge type="files"/>'
fi
 echo "$2</Layout>"
}

# funkcja generuje plik zgodny z menu.xml
# $1 katalog w którym znajduje się drzewo systemu menu (musi być zachowany po instalacji ⤸
menu)
generate_menu_xml() {
 echo '<Menu>
 <DefaultLayout>
  <Merge type="menus"/>
  <Merge type="files"/>
  <Separator/>
  <Menuname>all</Menuname>
 </DefaultLayout>'
print_dir "$1" " "
echo '</Menu>'
}

print_service_menus_action() {
 if [ -f "$1" ]; then
  action=`basename $1`
  action=${action%.desktop}
  echo "[Desktop Action $action]"
  echo "Exec=`grep '^Exec=' $1 | cut -f2- -d'='`"
  echo "Name=`grep '^Name=' $1 | cut -f2- -d'='`"
  echo "Icon=`grep '^Icon=' $1 | cut -f2- -d'='`"
  echo ""
```

```
  action_list="${action_list}${action};"
fi
}

generate_service_menus() {
 # dzieki ustawieniu zmiennej IFS na | read czyta jako osobne zmienne
 # pola rozdzielane | a nie białymi znakami
 pushd $1 > /dev/null
grep -v '^[ \t ]*#' "service_menus.cfg" | grep -v '^[ \t]*$' | while IFS="|" read name mime ⤸
icon apps; do
  file_name=`echo "${mime}_${name}.desktop" | tr '/* ' '-x_' | cut -c 1-128`
  if [ -e "$2/mymenu-$file_name" ]; then
   while true; do
    tmp_file_name=${file_name}_${RANDOM}
    if [ ! -e "$2/mymenu-$tmp_file_name" ]; then
     file_name=$tmp_file_name
     break
    fi
   done;
  fi
  : > "$2/mymenu-$file_name"
  action_list=""

  for app in $apps; do
   if [ "$app" = "SEPARATOR" ]; then
    action_list="${action_list}_SEPARATOR_;"
   elif [ -d "$app" ]; then
    for f in "$app"/*; do
     print_service_menus_action "$f" >> "$2/mymenu-$file_name"
    done
   else
    print_service_menus_action "$app" >> "$2/mymenu-$file_name"
   fi
  done;

  if [ ${#action_list} -ne 0 ]; then
   echo "[Desktop Entry]" >> "$2/mymenu-$file_name"
   echo "ServiceTypes=$mime" >> "$2/mymenu-$file_name"
   echo "Actions=${action_list}_SEPARATOR_" >> "$2/mymenu-$file_name"
   echo "Icon=$icon" >> "$2/mymenu-$file_name"
   echo "X-KDE-Priority=TopLevel" >> "$2/mymenu-$file_name"
   echo "X-KDE-Submenu=$name" >> "$2/mymenu-$file_name"
  fi
 done
 popd > /dev/null
}

# $1 katalog źródłowy    $2 katalog docelowy
install_menu() {
 # kopiowanie (gdy katalog źródłowy rożny od docelowego)
 if [ ! "`realpath $1`" -ef "`realpath $2 2> /dev/null`" ]; then
  rm -fr "$2"
  cp -r "$1" "$2"
fi

# instalacja ikon
 icon_path=`kde-config --expandvars --path icon | awk 'BEGIN {RS=":"} $0 ~ "^'$HOME'" {print ⤸
 $0}'`
rm -f "$icon_path"/mymenu_*
 for f in "$2"/icons/*; do
```

```
  ln -s -f "$f" "$icon_path"/mymenu-`basename $f`
done;

# generowanie menu
 menu_path=`kde-config --expandvars --path xdgconf-menu | awk 'BEGIN {RS=":"} $0 ~ "^'$HOME'" ⤸
 {print $0}'`
generate_menu_xml "$2" > "$menu_path/mymenu.xml"

# dodanie menu do menu
 menu_path="$menu_path/applications-kmenuedit.menu"
config_entry="<MergeFile>mymenu.xml</MergeFile>"
# gdy nie było dodane
 if ! grep $config_entry $menu_path > /dev/null; then
  tmp_file=`mktemp`
  #config_entry="<Menu><Name>Pakiety</Name>$config_entry</Menu>"
  # przy pomocy sed'a zastepujemy pierwsze wystąpienie <Menu> w wskazanym pliku
  sed -e '0,/<Menu>/ s#<Menu>#<Menu>'"$config_entry"'#' < $menu_path > $tmp_file
  # wykonanie TYLKO na pierwszym wystąpieniu w pliku uzyskujemy dzięki ograniczeniu
  # zakresu przetwarzanych linii od zerowej do wyrażenia regularnego
  if [ $? = 0 ]; then
   mv $menu_path $menu_path~
   mv $tmp_file $menu_path
  fi
 fi

# generowanie menu
 rm -f "`kde-config --localprefix`share/apps/konqueror/servicemenus"/mymenu-*
 generate_service_menus "$2" "`kde-config --localprefix`share/apps/konqueror/servicemenus"
}

# funkcja zwraca katalog w którym znajduje się uruchamiany skrypt
source_dir() {
 # ustalanie ścieżki skryptu
 sciezka=$0
 if [ ${sciezka:0:1} != "/" ]; then
  sciezka="$PWD/$0"
fi
 realpath `dirname $sciezka`
}

install_menu "`source_dir`" "$HOME/.config/mymenu"
kbuildsycoca
kdialog --msgbox 'Gotowe !!!'
```

Inne informacje

Przy zabawie z konfiguracją KDE przydatne może być polecenie `kbuildsycoca` przebudowywujące cache konfiguracji bez potrzeby przelogowywania się. Na koniec mała ciekawostka - jak wiadomo jedynym ograniczeniem co do nazw plików i katalogów w linuxie jest to aby nie zawierały znaku `/`, jednak w kde możemy uzyskać pozór tego znaku należy w nazwie zamiast niego umieścić `%2f` (uwaga lepiej taką nazwę pliku zrobić w trybie tekstowym bo w graficznym zostanie wstawiony podeójny `%` i efektu nie uzyskamy).

Wśród wielu standardowo rozpowszechnianych bibliotek/pluginów warto zwrócić uwagę na `libarts1-xine` umożliwiającą pokazywanie podglądu plików wideo w Konquerorze. W Sieci można znaleźć nie mało rozszerzeń dla kde (w dużej mierze korzystających z opisanych możliwości konfiguracyjnych) Biblioteki dla kio (umożliwiają korzystanie z zawartości archiwów, protokołów sieciowych itd jak normalnych ścieżek) dla: RAR[1835], 7zip[1836]. Dodatkowe wpisy menu usług dla: obsługi archiwów 7zip[1837], paczek deb[1838], montowania obrazów ISO[1839], otwierania jako inny użytkownik[1840] oraz różne inne użytki dla konquerora[1841] (kopiowanie ścieżek, zaznaczanie na podstawie wyrażeń regularnych, ustawianie obrazka jako ikony katalogu, ...). Dodatkowe wpisy dla przeglądania WWW w konquerorze

umożliwiające pobieranie filmików z youtoube[1842], a także patch[1843] na kdegraphics włączający podgląd i informacje exif dla plików graficznych typu raw.

Konqueror przy połączeniach SFTP, ma problem z obsługą kluczy typu ecdsa-sha2-nistp256. Objawem jest komunikat o zmienionym kluczu ("The host key for this server was not found, but another type of key exists. An attacker might change the default server key to confuse your client into thinking the key does not exist. Please contact your system administrator."). Klucze takie pojawiają się często w wyniku połączenia SSH na jakiś sprzęt wbudowany (r-pi, stb, ...). W takim wypadku należy usunąć klucz z .ssh/known_hosts, połączyć się Konquerorem przez SFTP, zaakceptować nowy klucz (do pliku dopisany zostanie klucz typu ssh-rsa).

Zobacz w Sieci: Przewodnik użytkownika KDE[1844].

Linki

Zobacz w Sieci (Debian): Debian[1845], Polish Debian Documentation Project[1846], Tworzymy paczkę[1847], Tworzenie własnej wersji Knoppiksa[1848], APT HOWTO[1849], Debian @ Wikibooks[1850], skrypty ułatwiające mirrorowanie Debiana[1851], chroot shell dla debiana[1852], Automatyzacja instalacji Debiana[1853].

Zobacz w Sieci (inne): g4u - Harddisk Image Cloning for PCs[1854], Pierwsze kroki ze Scribusem[1855], Recovering from file system corruption using TestDisk[1856], 6 terminali graficznych z jednego komputera[1857], GNU/Linux @ Wikibooks[1858], Linux Newbie Administrator Guide[1859], UnionFS[1860].

Konfiguracja środowiska pracy

Artykuł ten przedstawia system stanowiący kolekcję konfigów dla podstawowego oprogramowania składającego się na (graficzne) środowisko pracy:

- powłoka tekstowa - bash lub zsh
- manager ekranów tekstowych - tmux lub screen
- konsolowy edytor tekstowy - vim
- konsolowy przeglądarka WWW - w3m lub elinks albo links2
- konsolowy kalkluator - calc
- sporo innych narędzi shellowych nie wymagających specjalnej konfiguracji lub konfigurowanych w ramach ustawień powłoki - grep, ping, trace, awk, sed, ...

- manager okien, odpowiedzialny także za ogólnosystemowe skróty klawiaturowe - openbox
- panel, posiadający także moduł menagerera schowka - fbpanel + dodatkowe/zmodyfikowane pluginy
- klawiatura ekranowa - florence (conajmniej 0.6.0) lub kvkbd
- emulator terminala - xterm lub xfce4-terminal
- manager plików z obsługą podglądów, protokołów sieciowych itd - konqueror
- przeglądarka WWW - firefox lub konqueror
- odtwarzacz multimediów - vlc
- czytnik PDF - oklular lub xpdf
- przeglądarka plików graficznych - gwenview
- graficzny edytor tekstowy - kate

- klient poczty elektronicznej (SMTP, IMAP), grup newsowych (NNTP) - Claws Mail (graficzny), Cone (konsolowy)
- klient IM (XMPP/Jabber) - Psi-plus lub Pidgin (graficzne), Finch (konsolowy)
- klient IRC - Pidgin (graficzny), Finch (konsolowy)
- klient RSS - Claws Mail (graficzny), Canto i rsstail (konsolowe)
- klient SIP z wsparciem wideo, ipv6, wielu kont (register i proxy) oraz połączeń bezposrednich - linphone (graficzny i konsolowy)

System opiera się o skrypt instalujący wybrane zestawy oprogramowania z repozytoriów Debiana oraz skrypty i pliki konfiguracyjne z serwisu opcode. Ponaddto system zawiera narzędzie eksportu i importu zakładek dla firefoxa i konquerora ułatwiające utrzymanie wspólnej konfiguracji na wielu hostach. Konfigi dostępne są w formie archiwum tar[1861] (wersja rozpakowana[1862]). system jest w wersji beta

Pobierz skrypt instalacyjny[1863]

Zobacz także: zbiór prztydatnych ikonek[1864], moje łatki do fbpanel'a[1865].

Lista rzeczy do dopracowania itp

- dopracowanie konfigów narzędzi konsolowych
- testy i dopracowanie skryptu instalacyjnego

— coś w tary'u wykonującego po kliknięciu w jego ikonkę akcję - np. minimalizację do traya okien programów komunikacyjnych (patrz funkcje w autostart.sh)

WORKENV-INSTALL.SH

```
#!/bin/bash


#
#   LISTY PACZEK DEBIANOWYCH
#
PKG_LIST_11_shell_utils() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="narzędzia shellowe"
GRP_DEF="on"
PKGS="$PKGS bash apcalc bc gawk"
PKGS="$PKGS coreutils bsdutils bsdmainutils"
PKGS="$PKGS diffutils findutils grep sed tar util-linux gzip"
PKGS="$PKGS perl python bash-completion busybox manpages"
PKGS="$PKGS less file screen tmux vim-nox konwert whiptail"
PKGS="$PKGS bzip2 p7zip p7zip-full p7zip-rar gnupg2 gpgsm"
}

PKG_LIST_31_sys_utils_adm() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="narzędzia systemowe - administracyjne"
GRP_DEF="on"
PKGS="$PKGS lsof procps psmisc sudo locales"
PKGS="$PKGS sysstat iotop itop atop htop"
PKGS="$PKGS parted gdisk gnu-fdisk util-linux"
}

PKG_LIST_32_sys_utils_hwdiag() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="narzędzia systemowe - dignostyka sprzętu"
GRP_DEF="on"
PKGS="$PKGS pciutils usbutils lsscsi dmidecode"
PKGS="$PKGS sdparm smartmontools hddtemp hdparm"
}

PKG_LIST_33_sys_utils_hw() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="narzędzia systemowe - obsługa sprzętu"
GRP_DEF="on"
PKGS="$PKGS mdadm lvm2 dmsetup multipath-tools xfsprogs"
PKGS="$PKGS ipmitool lm-sensors pcmciautils"
PKGS="$PKGS debootstrap picocom"
PKGS="$PKGS gpm"
}

PKG_LIST_34_sys_utils_addional() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="narzędzia systemowe - dodatkowe (nfs, mail, gpm, ...)"
GRP_DEF="off"
PKGS="$PKGS bsd-mailx nullmailer rssh nfs-kernel-server"
}


PKG_LIST_41_net_conf() {
```

```
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="podstawowe pakiety obsługi sieci komuterowych"
GRP_DEF="on"
PKGS="$PKGS net-tools bridge-utils vlan iproute ethtool"
PKGS="$PKGS iputils-ping iputils-tracepath"
PKGS="$PKGS arptables ebtables iptables"
}

PKG_LIST_42_net_utils() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="programy sieciowe - ssh, przeglądarki www, ..."
GRP_DEF="on"
PKGS="$PKGS wget curl httrack w3m w3m-img links2"
PKGS="$PKGS rsync ntpdate bind9-host"
PKGS="$PKGS openssh-client openssh-server sshfs"
PKGS="$PKGS cone finch linphone-nogtk canto rsstail"
}

PKG_LIST_43_net_diag() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="narzędzia diagnostyczne do sieci komuterowych"
GRP_DEF="on"
PKGS="$PKGS tcptraceroute paris-traceroute mtr-tiny"
PKGS="$PKGS vnstat iftop"
PKGS="$PKGS arping arp-scan ndisc6"
PKGS="$PKGS netdiag"
PKGS="$PKGS tcpdump tcpflow ngrep nmap"
PKGS="$PKGS dnstop dnstracer dnsutils dnswalk"
PKGS="$PKGS telnet netcat netcat6 netcat-traditional socat picocom"
PKGS="$PKGS sipcalc ipcalc ipv6calc"
PKGS="$PKGS cutecom jpnevulator picocom"
}

PKG_LIST_44_net_wifi() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="obsługa sieci bezprzewodowych"
GRP_DEF="off"
PKGS="$PKGS wireless-tools wpasupplicant rfkill"
}

PKG_LIST_61_desktop() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="środowisko graficzne"
GRP_DEF="off"
PKGS="$PKGS xorg slim openbox fbpanel wmctrl xdotool"
PKGS="$PKGS feh xcompmgr florence"
}

PKG_LIST_62_desktop_multimedia() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="środowisko graficzne - multimedia"
GRP_DEF="off"
PKGS="$PKGS alsa-base alsa-utils"
PKGS="$PKGS vlc mpv"
PKGS="$PKGS xpdf"
}

PKG_LIST_63_desktop_kde() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="środowisko graficzne - konqueror i dodatki"
```

```
GRP_DEF="off"
PKGS="$PKGS konqueror dolphin mplayerthumbs kdegraphics-strigi-plugins"
PKGS="$PKGS konq-plugins kio-ftps kio-gopher kdemultimedia-kio-plugins kdepimlibs-kio-⤶
 plugins"
PKGS="$PKGS ark unrar unzip kate okteta okular okular-extra-backends gwenview kfind"
}

PKG_LIST_64_desktop_comunication() {
 GRP_NAME=${FUNCNAME#PKG_LIST_}
 GRP_DESC="środowisko graficzne - komunikacja"
GRP_DEF="off"
PKGS="$PKGS iceweasel ca-certificates"
PKGS="$PKGS claws-mail claws-mail-feeds-reader claws-mail-address-keeper claws-mail-fancy-⤶
 plugin claws-mail-multi-notifier"
PKGS="$PKGS claws-mail-smime-plugin claws-mail-pgpinline claws-mail-pgpmime claws-mail-⤶
 attach-remover claws-mail-tnef-parser"
PKGS="$PKGS psi-plus psi-plus-plugins psi-plus-plugin-psimedia linphone"
}

# SCHROOT ENVIROMENTS:
#
# # audio_video:
# # deb http://ftp.icm.edu.pl/pub/Linux/debian jessie main non-free contrib
# # deb ftp://ftp.icm.edu.pl/pub/linux/distributions/debian-multimedia jessie main non-free ⤶
contrib
# #   - audio:
#           ardour ardour3 audacity kwave rosegarden sweep timidity wavesurfer
# #   - video:
#           avidemux avidemux-cli cinelerra dvbcut kdenlive kino lives pitivi
# #   - tools:
#           ffmpeg mencoder ogmtools sox transcode transcode-utils mjpegtools
# #   - addionals players:
#           audacious mplayer
#
# # dtp_graphics_office:
# # deb http://ftp.icm.edu.pl/pub/Linux/debian jessie main
# #   - graphics
# #     > raster:
#           gimp kolourpaint4
# #     > vector:
#           inkscape dia librecad
# #     > 3d:
#           blender k3d freecad
# #     > tools:
#           graphicsmagick imagemagick netpbm exiv2 librsvg2-bin dvipng
# #   - dtp:
# #     > (La)TeX:
#           texlive tex-common texlive-base texlive-binaries texlive-luatex texlive-extra-utils ⤶
texlive-font-utils
#           texlive-latex-base texlive-latex-extra texlive-latex-recommended texlive-formats-⤶
extra lmodern tex-gyre
#           texlive-generic-extra texlive-generic-recommended texlive-lang-english texlive-lang-⤶
polish
#           texlive-math-extra texlive-pictures texlive-plain-extra texlive-publishers texlive-⤶
science
# #     > narzędzia:
#           ghostscript psutils poppler-utils ps2eps pdftk
#           xalan xmlstarlet libsaxonb-java blahtexml highlight
# #     > scribus:
#           scribus
```

```
# #    - office:
#              libreoffice-writer libreoffice-calc libreoffice-impress
#              libreoffice-math libreoffice-dmaths libreoffice-draw
#              aspell-en aspell-pl    myspell-en-us myspell-pl
#
# # engineering:
# # deb http://ftp.icm.edu.pl/pub/Linux/debian jessie main
# #    - electreonics:
#              geda geda-gschem geda-utils geda-xgsch2pcb pcb gerbv kicad
# #       > simulators:
#              oregano gnucap tkgate
# #       > programators:
#              avarice avrdude avrprog uisp
# #    - software developers:
#              gcc g++ clang gdb make cmake pkg-config
#              gcc-avr avr-libc gdb-avr
#              doxygen graphviz manpages-dev
# #       > repos:
#              bzr bzrtools git git-core subversion mercurial patch
# #    - other tools:
#              pythoncad gnuplot-x11 gnuplot5-x11
#              xutils-dev devscripts cpio gawk


#
#   OBSŁUGA INSTALACJI PACZEK DEBIANOWYCH
#
list_group() {
 set | \grep "PKG_LIST_$1[a-z0-9_]* (" | cut -f1 -d" " | sort
}

list_group2() {
 for tag in `list_group $1`; do
  $tag
  echo -n "$GRP_NAME \"$GRP_DESC\" $GRP_DEF  "
done
}

install_debian_soft() {
 if ! which whiptail; then
  a="?"
  while [ "$a" != "y" -a "$a" != "Y" -a "$a" != "" -a "$a" != "n" ]; do
  echo "Skrypt wymaga do poprawnej pracy pakietu whiptail czy chesz go zainstalować ? (Y/n)"
   read a
  done
  if [ "$a" != "n" ]; then
   aptitude --without-recommends install whiptail
  fi
 fi

# generacja okna dialogowego z listą grup do wyboru
 CMD="whiptail --separate-output --checklist \"Wybierz zestawy oprogramowania do instalacji:
 \" 0 0 0 -- `list_group2`"
unset PKGS
tag_list=$(eval $CMD 3>&1 1>&2 2>&3)

# sprawdzanie stanu pakietów
 for tag in $tag_list; do
   PKG_LIST_$tag
   echo -e "\033[01;33m$GRP_DESC\033[0m"
```

```
   for pkg in $PKGS; do
    if dpkg -s $pkg > /dev/null 2>/dev/null; then
     status="\033[01;32mYES\033[0m"
    else
     status="\033[01;31mNO \033[0m"
     IL="$IL $pkg"
    fi
    echo -e "$status  $pkg"
   done
   unset PKGS
done

# pytanie o instalację i ewentualna instalacja brakujących pakietów
 if [ "$IL" != "" ]; then
  a="?"
  while [ "$a" != "y" -a "$a" != "Y" -a "$a" != "" -a "$a" != "n" ]; do
   echo "zainstalować brakujące pakiety (Y/n)"
   read a
  done
  if [ "$a" != "n" ]; then
   aptitude --without-recommends install $IL
  fi
 fi
 unset IL
}


#
#   INSTALACJA RZECZY POBIERANYCH Z OPCODE
#

# $1 = install dir, $2 = base dir on server, $3 i kolejne = server filenames
# gdy $3 == "" $2 wskazuje archiwum tar do rozpakowania w $1
get_stuff_from_opcode() {
 url=http://www.opcode.eu.org/$2
 dst_dir=$1
 shift 2

mkdir -p "$dst_dir"

if [ "$1" = "tar:" ]; then
  shift 1
  for f in $@; do
   wget -nv -O - "$url/$f" | tar  -C "$dst_dir" --transform "s#^${f%.tgz}/##" -xzf -
   rmdir "$dst_dir/${f%.tgz}"
  done
 elif [ "$1" = "file:" ]; then
  shift 1
  for f in $@; do
   wget -nv -O "$dst_dir/$f" "$url/$f"
  done
 elif [ "$1" = "dir-file:" ]; then
  shift 1
  for f in $@; do
   wget -nv -O "$dst_dir/$f" "$url/$f/$f"

  done
 fi

for f in $@; do
```

```
    if [ "${f:$(( ${#f} - 3 ))}" = ".sh" ]; then
      mv "$dst_dir/$f" "$dst_dir/${f%.sh}"
      chmod +x "$dst_dir/${f%.sh}"
    fi
  done
}

install_soft4electronic() {
  get_stuff_from_opcode  ${1-/usr/local/bin/}  usage_and_config/soft4electronic  dir-file: \
    gsch2pdf.sh pcb2pdf.sh dia2pdf.sh \
    gsch2pcb-gen_sub_pcb.sh build_pcb_version.sh \
    change_size_in_pcb.sh expr2pla.sh
}

install_soft4electronic_config() {
  if which pcb > /dev/null; then
    get_stuff_from_opcode "$HOME/.pcb" "usage_and_config/soft4electronic" file: "pcb.inc"
    ln -s "$HOME/.pcb/pcb.inc" "$HOME/.pcb/user-config.inc"
    get_stuff_from_opcode "$HOME/.pcb/elements" "usage_and_config/soft4electronic" tar: "pcb-⤸
    elements.tgz"
  fi

  if which dia > /dev/null; then
    get_stuff_from_opcode "$HOME/.dia/sheets" "usage_and_config/soft4electronic" file: ⤸
    "Assorted.sheet"
  fi

  if which gschem > /dev/null; then
    get_stuff_from_opcode "$HOME/.gEDA/symbole/opcode.eu.org" ⤸
    "usage_and_config/soft4electronic" tar: "gschem-symbole.tgz"
    wget -nv -O "$HOME/.gEDA/commonrc" "http://www.opcode.eu.⤸
    org/usage_and_config/soft4electronic/geda-commonrc"
    wget -nv -O "$HOME/.gEDA/gschemrc" "http://www.opcode.eu.⤸
    org/usage_and_config/soft4electronic/geda-gschemrc"
    ln -s "$HOME/.gEDA/commonrc" "$HOME/.gEDA/gafrc"
    wget -nv -O "$HOME/.pcb/preferences" "http://www.opcode.eu.⤸
    org/usage_and_config/soft4electronic/pcb-preferences"
  fi
}


install_misc_stuff() {
  get_stuff_from_opcode  ${1-/usr/local/bin}  ⤸
  "usage_and_config/multimedia/rtv_recorder_and_transmiter" file: rtv.sh
  get_stuff_from_opcode  ${1-/usr/local/bin}  "usage_and_config/multimedia" dir-file: tv.vlc.⤸
  sh set_station.sh

  get_stuff_from_opcode  ${1-/usr/local/bin}  "usage_and_config/toolbox4computer" dir-file: ⤸
  alarm.sh zegar.sh
}

install_misc_config() {
  if which latex > /dev/null; then
    get_stuff_from_opcode "$HOME/.tex" "archiwum/ROZNOSCI" tar: tex.tgz
  fi
}

prepare_icons() {
  mkdir -p "$HOME/.local/share/icons"
  while read src dst; do
```

```
  dst="$HOME/.local/share/icons/$dst.${src##*.}"
  if [ -e "$src" ]; then
   if [ -e "$dst" ]; then
    if [ "$1" = "-f" ]; then
     rm "$dst"
     ln -s "$src" "$dst"
    fi
   else
    ln -s "$src" "$dst"
   fi
  fi
  # taka konstrukcja umożliwia wywołanie prepare_icons
  # z kilkoma wskazaniami tego samego celu
  # - wygrywa ostatnie istniejace źródło
  #
  # pozwala to na tworzenie list ikon przewidujących
  # kilka wariantów dostępnych zestawów ikon
 done
}

prepare_icons_for_fbpanel() {
 echo "/usr/share/icons/oxygen/64x64/apps/preferences-system-login.png fbpanel-actions
 /usr/share/icons/oxygen/64x64/categories/applications-other.png fbpanel-apps
 /usr/share/icons/oxygen/64x64/places/folder-favorites.png fbpanel-bookmark
 /usr/share/icons/oxygen/64x64/places/favorites.png fbpanel-bookmark-file
 /usr/share/icons/oxygen/64x64/places/folder-favorites.png fbpanel-bookmark-subdir
 /usr/share/icons/oxygen/64x64/apps/accessories-calculator.png fbpanel-calculator
 /usr/share/icons/oxygen/64x64/actions/edit-paste.png fbpanel-clip
 /usr/share/icons/oxygen/64x64/actions/application-exit.png fbpanel-screen-disable
 /usr/share/icons/oxygen/64x64/actions/system-shutdown.png fbpanel-halt
 /usr/share/icons/hicolor/64x64/apps/iceweasel.png fbpanel-iceweasel
 /usr/share/icons/hicolor/64x64/apps/konqueror.png fbpanel-konqueror
 /usr/share/icons/oxygen/64x64/actions/system-lock-screen.png fbpanel-screen-lock
 /usr/share/icons/oxygen/64x64/actions/system-log-out.png fbpanel-logout
 /usr/share/icons/oxygen/64x64/places/folder-txt.png fbpanel-notes
 /usr/share/icons/oxygen/64x64/mimetypes/text-x-generic.png fbpanel-notes-file
 /usr/share/icons/oxygen/64x64/places/folder-documents.png fbpanel-notes-subdir
 /usr/share/icons/oxygen/64x64/actions/system-reboot.png fbpanel-reboot
 /usr/share/icons/oxygen/32x32/actions/view-refresh.png fbpanel-refresh
 /usr/share/icons/oxygen/64x64/actions/system-run.png fbpanel-run
 /usr/share/icons/oxygen/64x64/apps/utilities-terminal.png fbpanel-terminal
 /usr/share/icons/hicolor/64x64/apps/kate.png fbpanel-text-editor" | prepare_icons
}


install_configs_from_workenv() {
 tmp_dir=`mktemp -d`
get_stuff_from_opcode "$tmp_dir" "usage_and_config/operating_systems_config/linux/workenv" ⤸
tar: workenv.tgz

(
 cd "$tmp_dir"
echo "PODSTAWOWE KONFIGI TRYBU TEKSTOWEGO"
echo "=================================="
for p in shell/*; do
  f=`basename $p`
  \mv "$p" "$HOME/.$f"
done
```

```
echo ""
echo "PODSTAWOWE KONFIGI XWINDOW"
echo "=========================="
if which openbox > /dev/null; then
  mkdir -p "$HOME/.config/openbox"
  for p in xwindow_base/openbox/*; do
   f=`basename "$p"`
   [ -d "$HOME/.config/openbox/$f" ] && rm -fr "$HOME/.config/openbox/$f"
   \mv "$p" "$HOME/.config/openbox/$f"
  done
 fi

if which fbpanel > /dev/null; then
  mkdir -p "$HOME/.config/fbpanel"
  \mv xwindow_base/fbpanel "$HOME/.config/fbpanel/default"
  prepare_icons_for_fbpanel
fi

if which xfce4-terminal > /dev/null; then
  mkdir -p "$HOME/.config/Terminal"
  \mv xwindow_base/xfce4-terminalrc "$HOME/.config/Terminal/terminalrc"
fi

for p in xwindow_base/{gtkrc-2.0,gtkrc-2.0.mine,Xdefaults}; do
  f=`basename $p`
  \mv "$p" "$HOME/.$f"
done


echo ""
echo "KONFIGURACJA ELEMENTÓW KDE"
echo "=========================="
if which konqueror > /dev/null; then
  ( cd ./xwindow_kde/konqueror; bash ./configure.sh; )
  mkdir -p "$HOME/.bin"
  \mv xwindow_kde/scripts/* "$HOME/.bin"
fi
 which kate > /dev/null && ( cd ./xwindow_kde; bash ./kate.sh )


echo ""
echo "KONFIGI APLIKACJI"
echo "================="
if which claws-mail > /dev/null; then
  mkdir -p "$HOME/.claws-mail"
  \mv xwindow_other/claws-mail/* "$HOME/.claws-mail"
fi

if which psi-plus > /dev/null; then
  mkdir -p "$HOME/.config/Psi+/profiles/default"
  \mv xwindow_other/psi/* "$HOME/.config/Psi+/profiles/default"
fi

if [ -d /usr/share/pixmaps/linphone/ ]; then
  bdir=/usr/share/pixmaps/linphone/
  [ -e /$bdir/linphone_new.png ] || sudo mv /$bdir/linphone.png /$bdir/linphone_new.png
  sudo ln -fs /$bdir/linphone2-57x57.png /$bdir/linphone.png
fi

if which vlc > /dev/null; then
```

```
  mkdir -p "$HOME/.config/vlc"
  \mv xwindow_other/vlc/* "$HOME/.config/vlc"
fi

if which iceweasel > /dev/null; then
  echo "ICEWEASEL"
  echo "1. Install: keyconfig.xpi (See: http://forums.mozillazine.org/viewtopic.php?p=3042712)⤸
  "
  echo "2. add to prefs.js:"
  cat xwindow_other/firefox-prefs.js
fi

if which florence > /dev/null; then
  mkdir -p "$HOME/.config/vlc"
  \mv xwindow_other/florence/* "$HOME/gconf/apps/florence/"
fi


echo ""
echo "PLIKI KONFIGURACYJNE NIE ZARZĄDZANE SKRYPTEM"
echo "============================================"
ls -1d ~/.bashrc.local ~/.config/openbox/autostart-local.sh 2>/dev/null
ls -1d ~/.claws-mail/ ~/.cone/ 2>/dev/null
ls -1d ~/.linphonerc 2>/dev/null
ls -1d ~/.gntrc ~/.purple ~/.psi/profiles/*/config.xml ~/.psi/profiles/*/options.xml 2>⤸
 /dev/null
ls -1d ~/.w3m/config ~/.elinks ~/.links2 2>/dev/null
ls -1d ~/.gnupg ~/.ssh/config 2>/dev/null

echo "... i inne"
)

rm -fr "$tmp_dir"
}


a="?"
while [ "$a" != "q" ]; do
 echo "wybierz akcję do wykonania:"
echo "1 - aktualizacja skryptu"
echo "2 - instalacja pakietów Debianowych"
echo "3 - instalacja skryptów itp z opcode.eu.org w /usr/local/bin"
echo "4 - instalacja skryptów itp z opcode.eu.org w ~/.bin"
echo "5 - instalacja konfiguracji linux-workenv z opcode.eu.org"
echo "6 - instalacja pozostałej konfiguracji opcode.eu.org"
echo "q - wyjście"
read a
if [ "$a" = "1" ]; then
  tmp_file=`mktemp`
  if wget -O "$tmp_file" http://www.opcode.eu.⤸
  org/usage_and_config/operating_systems_config/linux/workenv/workenv-install.sh/workenv-⤸
  install.sh; then
   mv "$tmp_file" "$0"
   chmod 0755 "$0"
   exec "$0"
  fi
 elif [ "$a" = "2" ]; then
  install_debian_soft
elif [ "$a" = "3" ]; then
  install_soft4electronic
```

```
  install_misc_stuff
elif [ "$a" = "4" ]; then
  install_soft4electronic $HOME/.bin
  install_misc_stuff $HOME/.bin
elif [ "$a" = "5" ]; then
  install_configs_from_workenv
elif [ "$a" = "6" ]; then
  install_soft4electronic_config
  install_misc_config
else
  continue
 fi
 echo "ACTION $a DONE"
done
```

Poradnik konfiguracyjny POSIX - serwerowe

Sieciowe bazy danych użytkowników - NIS, LDAP

Tradycyjnie udostpnianiem danych użytkowników dla wielu maszyn zajmował się NIS[1866]. Współcześnie jego miejsce coraz częściej zajmuje LDAP[1867].

Konfiguracja oby tych machanizmów może być oparta na pliku `/etc/nsswitch.conf` w którym określamy źródła pochodzenia (takie jak file, nis, ldap, dns, …) dla różnych usług informacyjnych (baza hostów, baza protokołów, baza użytkowników, …). Możliwe jest także wykorzystanie PAM do konfiguracji tych mechanizów.

O ile sam LDAP pozwala na zagnieżdzanie grup w grupach itp to jest to ciężko zastosować do UNIXowego modelu grup uzytkowników, jednak w przypadku takiej potrzeby można stworzyć narzędzie zamieniające taką zagnierzoną strukturę na płaską listę wielu grup do których przynależy użytkownik.

Bazę LDAP możemy przeszukiwać za pomocą komendy `ldapsearch`, natomiast pełną informację np. o bazie passwd możemy uzyskać poprzez `getent passwd` (jest to odpowiednik znanego z systemów NIS `ypcat`).

Zobacz w Sieci: LDAP/NSS - Debian Wiki[1868], LDAP/PAM - Debian Wiki[1869]

Instalacja serwera LDAP

Jeżeli mamy ustawiony zbyt wysoki priorytet debconfa to instalator pakietu serwera LDAP (slapd) użyje nazwy domeny (bez nazwy serwera) do utworzenia suffix'u zakładanej bazy danych. Aby użyć czegoś innego najprościej użyć `dpkg-reconfigure slapd`.

Utworzona przy instalacji baza danych nie posiada struktury wykorzystywanej do uzyskiwania informacji o użytkownikach przez NSS oraz PAM. Można utworzyć ją manualnie lub wykorzystać Ldap Account Manager, który utworzy ją przy pierwszym zalogowaniu.

Konfiguracja Ldap Account Manager

Ldap Account Manager jest wygodnym interfejsem web-owym do zarządzania bazą użytkowników opartą o LDAP. Logowanie do systemu odbywa się przy pomocy konta i hasła administracyjnego LDAP (jest od razu logowaniem do serwera LDAP).

Przed zalogowaniem należy skonfigurować system do pracy ze swoim serwerem LDAP. W tym celu łączymy się z stroną konfiguracyjną LAM dostępną pod adresem http://nazwa.lub.IP.serwera/lam/templates/config/index.php i w sekcji dotyczącej profili serwerowych ("server profiles"):

- ustawiamy adres serwera (jeżeli LAM jest na innym serwerze niż LDAP)
- ustawiamy "Tree suffix" na zgodny z konfiguracją naszego serwera (można sprawdzić w `/etc/ldap/slapd.conf` - opcja `suffix`, jest to domena podawana w konfiguracji serwera LDAP w zapisie ldapowym)
- dostosowujemy do naszych potrzeb typy kont i moduły obsługiwane przez LAM. Uwaga: dla każdego z typów kont należy ręcznie zmienić standardowy surfix (dc=my-domain,dc=com) na taki który my używamy

— dostosowujemy zakresy UIDów i GIDów do naszych potrzeb (ważne aby nie dochodziło do konfliktów z lokalnymi konfiguracjami)
— edytujemy listę kont uprawnionych do korzystania z LAM (ważne aby tu podać poprawnego usera z prawami administracyjnymi - np. podanego jako `rootdn` w `/etc/ldap/slapd.conf`
— zmieniamy hasło dostępowe do tej konfiguracji

Warto także ustawić odpowiednie wymagania dotyczące haseł oraz zmienić hasło dostępu do sekcji dotyczącej ustawień globalnych. W obu wypadkach domyślnym hasłem dostępowym jest lam

Po tych zmianach należy zalogować się do LAM. System zaproponuje utworzenie brakującej struktury dla wybranych typów kont - zgadzamy się. Gdy utworzenie struktury zostanie wykonane, możemy przystąpić do zakładania grup i kont.

Administracja kontami LDAPowymi

```
URL="ldap://adres.of.ldap.serwer/"
BASEDN="dc=opcode,dc=eu,dc=org"
BINDDN="uid=MYADMINLOGIN,ou=People,$BASEDN"
# dodanie usera:
        ldapadd -xW -H "$URL" -D "$BINDDN" -x -W -f dodawanie.ldif
# dodanie do grupy lub inne modyfikacje:
        ldapadd -xW -H "$URL" -D "$BINDDN" -f modyfikacja.ldif
# zmiana hasla:
        ldappasswd -xW -H "$URL" -D "$BINDDN" -S "uid=LOGIN,ou=People,$BASEDN"
# sprawdzenie dzialania hasla:
        ldapsearch -xW -H "$URL" -b "$BASEDN" -D "uid=LOGIN,ou=People,$BASEDN"
# wyszukiwanie:
        ldapsearch -H "$URL" -b "$BASEDN" -x
```

Przyklad pliku ldif do zakladania uzytkownika:

```
dn: uid=LOGIN,ou=People,dc=vls,dc=icm,dc=edu,dc=pl
objectClass: ldapPublicKey
objectClass: shadowAccount
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
shadowWarning: 10
shadowInactive: 10
shadowMin: 1
shadowMax: 365
shadowLastChange: 14673
homeDirectory: /home/LOGIN
loginShell: /bin/false
uid: LOGIN
cn: IMIE NAZWISKO
uidNumber: 30001
sn: NAZWISKO
givenName: IMIE
mail: ADRES@MAILOWY
gidNumber: 3003
userPassword: {MD5}da4427d9e3dbb96a7fe4053a4bfb21dd
```

Przyklad pliku ldif do dodania uzytkownika do grupy:

```
dn: cn=NAZWAGRUPY,ou=Group,dc=vls,dc=icm,dc=edu,dc=pl
changetype: modify
add: memberUid
memberUid: LOGIN
```

Klienci bazy LDAP

Autoryzacja kont systemowych w LDAP z użyciem PAM i NSS

W przypadku korzystania z LDAP do autoryzacji użytkowników systemowych wymagane jest umieszczenie odpowiedniej konfiguracji (wskazującego źródło informacji autoryzacyjnych LDAP) w plikach `/etc/nss-ldapd.conf` (lub `/etc/libnss-ldap.conf`) oraz `/etc/pam_ldap.conf`. Jako że pliki te są łudząco podobne do siebie należy uważać aby nie powstawały rozbieżności między nimi - niespójność danych w tych plikach grodzi dziwnymi i trudnymi do diagnozowania błędami) oraz odpowiedniej konfiguracji samego pam w plikach `/etc/pam.d/*` np:

```
account    required     pam_ldap.so
auth       sufficient   pam_ldap.so use_first_pass
password   sufficient   pam_ldap.so use_first_pass
session    optional     pam_ldap.so
```

Konfiguracja krok po kroku:

1. `aptitude install libnss-ldapd libpam-ldap`
2. w trakcie konfiguracji podajemy (dwu krotnie - osobno dla nss i osobno dla pam) poprawny URI serwera LDAP oraz bazowe DN dla wyszukiwania (zwyczajowo jest to nasza domena zapisywana w postaci kolejnych `dc=` składających się z kolejnych członów domeny rozdzielanych kroplami - np. dla subdomena.example.net będzie to dc=subdomena,dc=example,dc=net)

   aby nie musieć podawać danych autoryzacyjnych LDAP (gdy nasz serwer odpowiada na niezautoryzowane zapytania) na pytania o zezwolenie lokalnemu root'owi na dostęp administracyjny oraz o to czy LDAP wymaga autoryzacji odpowiadamy przecząco
3. W tym momencie komenda `grep -v '^#' /etc/*ldap*.conf  | grep -v '^[^:]*:$'` powinna zwracać coś na kształt:
   ```
   /etc/nss-ldapd.conf:uid nslcd
   /etc/nss-ldapd.conf:gid nslcd
   /etc/nss-ldapd.conf:uri ldap://mojserwerldap/
   /etc/nss-ldapd.conf:base dc=subdomena,dc=example,dc=net
   /etc/pam_ldap.conf:base dc=subdomena,dc=example,dc=net
   /etc/pam_ldap.conf:uri ldap://mojserwerldap/
   /etc/pam_ldap.conf:ldap_version 3
   /etc/pam_ldap.conf:pam_password crypt
   ```
   Polecam także dodanie do pliku `/etc/pam_ldap.conf` ograniczenia czasu oczekiwania na odpowiedź serwera LDAP:
   ```
   timelimit 10
   bind_timelimit 10
   bind_policy soft
   ```
4. W przypadku nowszych systemów odpowiadamy na pytania dotyczące modyfikacji konfiguracji PAM aby używać autoryzacji UNIXowej i LDAP w przypadku starszych samodzielnie modyfikujemy pliki `/etc/pam.d/common-*`:
   - `common-account` zastępujemy następującą zawartością:
     ```
     account [success=2 new_authtok_reqd=done default=ignore]        pam_unix.so
     account [success=1 default=ignore]      pam_ldap.so
     account requisite                       pam_deny.so
     account required                        pam_permit.so
     account required                        pam_access.so
     ```
   - `common-auth` zastępujemy następującą zawartością:
     ```
     auth    [success=2 default=ignore]      pam_unix.so nullok_secure
     auth    [success=1 default=ignore]      pam_ldap.so use_first_pass
     auth    requisite                       pam_deny.so
     auth    required                        pam_permit.so
     ```
   - w `common-password` dodajemy:
     ```
     password    sufficient    pam_ldap.so
     password    required      pam_deny.so
     ```
   - w `common-session` dodajemy:
     ```
     session     required      pam_limits.so
     session     optional      pam_ldap.so
     ```

Opcjonalnie możemy dodać `session required pam_mkhomedir.so skel=/etc/skel umask=0022` aby automatycznie tworzyć katalogi domowe

5. Wpisy ldap przy wybranych przez nas pozycjach w `/etc/nsswitch.conf` powinny zostać dodane automatycznie, jeżeli nie robimy to ręcznie.
6. W `/etc/security/access.conf` umieszczamy wpis ograniczający prawo logowania do użytkowników znajdujących się w podanych grupach: `-:ALL EXCEPT root grupaAgrupaB:ALL EXCEPT LOCAL`

Klucze publiczne SSH z LDAP

W przypadku korzystania z SSH dostęp w oparciu o hasło PAMowe to nie wszystko. Wielu użytkowników wręcz preferuje dostęp oparty o klucze publiczne SSH przechowywane standardowo w `$HOME/.ssh/authorized_keys`. Kuszącym jest możliwość umieszczania tych kluczy również w bazie LDAP (dotyczy to zwłaszcza systemów mających centralną autoryzacje opartą o LDAP, a nie mających wspólnego `/home`). Istnieje stosowny patch do serwera SSH umożliwiający taką integrację - openssh-lpk[1870]. Niestety w chwili obecnej nie jest on włączony ani do samego serwera OpenSSH ani do paczki Debianowej, zatem konieczne jest jego samodzielne nałożenie. W tym celu:

```
aptitude install devscripts libldap-dev
aptitude build-dep openssh-server
svn checkout http://openssh-lpk.googlecode.com/svn/trunk/ openssh-lpk-read-only
apt-get source openssh-server
cd openssh-5.1p1/
# dodac w debian/rules w okolicy innych confflags:
# confflags += --with-libs="-lldap" --with-ldflags="-L/usr/lib"
↪ --with-cppflags="-I/usr/include -DWITH_LDAP_PUBKEY"
vi debian/rules
patch -p1 < ../openssh-lpk-read-only/patch/contrib/contrib-openssh-lpk-5.1p1-0.3.10.patch
debchange -i "openssh-lpk path"
dpkg-buildpackage -rfakeroot
```

Użycie patcha sprowadza się do dodania do `/etc/ssh/sshd_config`:

```
UseLPK yes
LpkLdapConf /etc/pam_ldap.conf
LpkForceTLS no
```

Po stronie serwera LDAP konieczne jest pobranie i zainstalowanie odpowiedniego schema:

```
cd /etc/ldap/schema && wget
↪ http://openssh-lpk.googlecode.com/files/openssh-lpk_openldap.schema
echo "include         /etc/ldap/schema/openssh-lpk_openldap.schema" >>
↪ /etc/ldap/slapd.conf
/etc/init.d/slapd restart
```

Autoryzacja HTTP w Apache2 w oparciu o LDAP

Aby Apache2 mógł autoryzować dostęp w oparciu o LDAP konieczne jest włączenie stosownych modułów (poprzez utworzenie dowiązań do odpowiednich plików z `/etc/apache2/mods-available/` w `/etc/apache2/mods-enabled/`: auth_basic, authn_file, authnz_ldap, authz_default, authz_groupfile, authz_host i authz_user

Następnie do ogólnej konfiguracji można dodać opcje takie jak:

```
LDAPSharedCacheSize 500000
LDAPCacheEntries 1024
LDAPCacheTTL 600
LDAPOpCacheEntries 1024
LDAPOpCacheTTL 600
<Location /ldap-status>
        SetHandler ldap-status
        Order deny,allow
        Deny from All
        Allow from 127.0.0.1
</Location>
```

Natomiast samą autoryzację włączmy wpisem typu:

```
<Directory /var/www/tajne_dane>
        AuthType Basic
        AuthName "Sorry - autoryzacja LDAP"
        AuthBasicProvider ldap
        AuthLDAPURL
 ↪  ldap://adres.naszego.serwera.ldap/ou=People,dc=subdomena,dc=example,dc=net?uid
        # ?uid oznacza w powyzszym że login jest w polu bazy LDAP o nazwie uid
        # możemy wymagać tylko posiadania konta w LDAPie:
        #Require valid-user
        # lub członkowstwa w podanej grupie (w tym wypadku wwwACL):
        Require ldap-group cn=wwwACL,ou=Group,dc=subdomena,dc=example,dc=net
        AuthLDAPGroupAttribute memberUid
        AuthLDAPGroupAttributeIsDN off
</Directory>
```

Autoryzacja w MediaWiki

Aby MediaWiki korzystało do autoryzacji użytkowników z bazy LDAP należy w `LocalSettings.php` umieścić wpisy:

```
require_once( "/usr/share/mediawiki-extensions/LdapAuthentication.php" );
$wgAuth = new LdapAuthenticationPlugin();
// podstawowa konfiguracja
$wgLDAPDomainNames = array( "JakasNazwaGrupyAutoryzacyjnej" );
$wgLDAPServerNames = array( "JakasNazwaGrupyAutoryzacyjnej" => "adres.naszegoserwera" );
$wgLDAPBaseDNs = array( "JakasNazwaGrupyAutoryzacyjnej"=>"dc=subdomena,dc=example,dc=net"
 ↪  );
$wgLDAPSearchStrings = array( "JakasNazwaGrupyAutoryzacyjnej" =>
 ↪  "uid=USER-NAME,ou=people,dc=subdomena,dc=example,dc=net" );
$wgLDAPEncryptionType = array( "JakasNazwaGrupyAutoryzacyjnej"=>"clear" );
// wymog czlonkowstwa w grupie
$wgLDAPRequiredGroups = array( "JakasNazwaGrupyAutoryzacyjnej"=>array(⌋
 ↪  "cn=wikiACL,ou=group,dc=subdomena,dc=example,dc=net")
 ↪  );
$wgLDAPGroupObjectclass = array( "JakasNazwaGrupyAutoryzacyjnej"=>"posixGroup" );
$wgLDAPLowerCaseUsername = array( "JakasNazwaGrupyAutoryzacyjnej"=>"true" );
$wgLDAPGroupAttribute = array( "JakasNazwaGrupyAutoryzacyjnej"=>"memberUid" );
//$wgLDAPGroupNameAttribute = array( "JakasNazwaGrupyAutoryzacyjnej"=>"cn" );
//$wgLDAPGroupUseFullDN = array( "JakasNazwaGrupyAutoryzacyjnej"=>false );
//$wgLDAPGroupSearchNestedGroups = array( "JakasNazwaGrupyAutoryzacyjnej"=>false );
```

Autoryzacja użytkowników Mantis'a w oparciu o LDAP

w Konfiguracji Mantisa w pliku `config_local.php`:

```
$g_send_reset_password = OFF;
$g_allow_signup = OFF;
$g_login_method = LDAP;
$g_ldap_protocol_version = 3;
$g_ldap_server = 'ldap://adres.naszego.serwera.ldap';
$g_ldap_port = '389';
#$g_ldap_bind_dn = 'cn=admin,dc=subdomena,dc=example,dc=net';
#$g_ldap_bind_passwd = 'root';
$g_ldap_root_dn = 'ou=People,dc=subdomena,dc=example,dc=net';
$g_ldap_group_dn = 'cn=mantisACL,ou=group,dc=subdomena,dc=example,dc=net';
#$t_ldap_memberuid_field
#$g_ldap_organization = '';
```

```
#$g_ldap_uid_field = 'uid';
```

Aby mantis automatycznie tworzył (swoje) konta dla użytkowników autoryzowanych w oparciu o LDAP konieczne jest zaaplikowanie łaty mojego autorstwa dostępnej jako załącznik do stosownego zgłoszenia - 0011470: auto-create mantis account for LDAP users - MantisBT[1871]. Jest to niezbędne aby użytkownicy LDAPowi mieli dostęp do mantisa bez ręcznego zakładania dla nich kont w Mantisie.

Aby działała autoryzacja w oparciu o grupie konieczne jest zaaplikowanie łaty mojego autorstwa dostępnej jako załącznik do stosownego zgłoszenia - 0011488: LDAP group based autentication - MantisBT[1872]. Alternatywnym rozwiązaniem jest 0008471: Add feature to check required LDAP attribute for authentication - MantisBT[1873].

LDAP password change on web page

Jest to prosty skrypt umożliwiający zmianę hasła LDAPowego przy pomocy strony WWW.
POBIERZ[1874]

passwd.php

```
<?php // KONFIGURACJA:
 $LDAP_SERVER = "ldap://adres.of.ldap.serwer";
 $LDAP_PORT = "389";
 $USERNAME_FIELD = "uid";
 $USERNAME_SURFIIX = "ou=People,dc=opcode,dc=eu,dc=org";
?>
<html>
 <head>
  <title>Change LDAP password</title>
 </head>
 <body>
 <h1 style="text-align:center">Change LDAP password</h1>
<?php if ($_POST['action'] != "change_passwd") { ?>
 <form method="post" action="<?php echo $_SERVER['SCRIPT_NAME'] ?>"><center><table>
  <tr>
   <td><label for="wpPassword">Username:</label></td>
   <td><input type="text" name="username" /></td>
  </tr>
  <tr>
   <td><label for="wpPassword">Password:</label></td>
   <td><input type="password" name="password" /></td>
  </tr>
  <tr>
   <td><label for="wpPassword">New password:</label></td>
   <td><input type="password" name="new_password_1" /></td>
  </tr>
  <tr>
   <td><label for="wpPassword">Retype new password:</label></td>
   <td><input type="password" name="new_password_2" /></td>
  </tr>
  <tr><td colspan="2">
   <input type="submit" value="OK" />
   <input type="hidden" name="action" value="change_passwd" />
  </td></tr>
 </table></center></form>
<? } else {
 if ($_POST['new_password_1'] == "") {
  echo "<p style=\"color: red\">Sorry, empty passwords do not allowed ... Password unchanged<
  /p>";
 } else if ($_POST['new_password_2'] != $_POST['new_password_1']) {
  echo "<p style=\"color: red\">Sorry, passwords do not match ... Password unchanged</p>";
 } else {
```

```
  $conn = @ldap_connect ( $LDAP_SERVER, $LDAP_PORT );
  if ($conn) {
   ldap_set_option( $conn, LDAP_OPT_PROTOCOL_VERSION, 3 );
   $user_rdn = "$USERNAME_FIELD=" . $_POST['username'] . ",$USERNAME_SURFIIX";

   $bind = @ldap_bind( $conn, $user_rdn, $_POST['password'] );
   if ($bind) {
    $password_hash = '{CRYPT}' . crypt($_POST['new_password_1'], '');
    $attrs = array('userpassword' => $password_hash);

    if (ldap_mod_replace($conn, $user_rdn, $attrs)) {
     echo "<p style=\"color: green\">Password changed !!!</p>";
    } else {
     echo "<p style=\"color: red\">Error changing passwords ... Password unchanged</p>";
    }
   } else {
    echo "<p style=\"color: red\">Error login to LDAP server ... Please check username or
    password</p>";
   }
  } else {
   echo "<p style=\"color: red\">Error connecting to LDAP server ... Please try again later<
   /p>";
  }
 }
} ?>
 </body>
</html>
```

Tworzenie "instalacji" postgres'a

Niekiedy potrzebujemy utworzenia dodatkowej instancji bazy danych postgres. w tym celu należy:

1. utworzyć użytkownika (user-db) który będzie penem i władcą tej instancji bazy danych
2. utworzyć katalog (/var/postgres-mydb) w którym będziemy trzymali naszą bazę, jego właścicielem musi być user-db
3. zainicjalizować bazę w tym katalogu komendą `initdb` (uruchomioną z pod user-db - tak samo będzie nazywał się użytkownik mający prawa administracyjne w naszej nowej bazie danych)
4. uruchomić jako user-db postgres'a `postgres -D /var/postgres-mydb` lub `pg_ctl -D /var/postgres-mydb -l logfile start`
5. bazy danych i użytkowników może tworzyć user-db poprzez wywołania komend `createdb` i `createuser`

Do wykonywania zapytań SQLowych może posłużyć narzędzie `psql`. Warto w nim zwócić uwagę na komendy `\l` i `\d` listujące odpowiednio bazy danych oraz tabele w aktualnej bazie danych (więcej informacji `\?`). Do kasowania użytkowinów i baz służą komendy `dropuser` i `dropdb`. Jeżeli mamy problem z kasowaniem bazy z powodu używania przez innego usera (ERROR: database ... is being accessed by other users) przydatne może się okazać wyświetlenie aktywności na tej bazie poprzez zapytanie SQLowe: `select * from pg_stat_activity where datname='NAZWABAZY';`

IPMI

IPMI[1875] jest specyfikacją komunikacji systemu zarządzającego sprzętem z interfejsem administracyjnym. Aby korzystać z pod Linuxa z dobrodziejstw tego wynalazku należy posiadać płytę główną (ew. inne urządzenie) wspierające ten protokół oraz zainstalować OpenIPMI i ipmitool. IPMI umożliwia m.in. (zdalny) podgląd informacji z sensorów, wyłączenie, włączenie czy też reset zasilania, a także zdalny dostęp do konsoli.

Lokalne używanie ipmi wymaga załadowania modułów ipmi_devintf i ipmi_si, zainstalowania wspomnianego openipmi oraz połączenia się przy pomocy komendy `ipmitool -I open -H 192.168.0.102 shell`. Możemy wtedy dokonać konfiguracji dostępu sieciowego:

```
# w testowanej płycie były dwie karty sieciowe z których mógł korzystać IPMI
# - identyfikowane w nim jako interfejsy 6 i 7
lan set 6 ipaddr 192.168.0.100
lan set 6 netmask 255.255.0.0
lan set 6 ipsrc static
lan set 6 auth ADMIN PASSWORD
lan set 6 auth OPERATOR PASSWORD
lan set 6 auth USER PASSWORD
lan set 6 auth CALLBACK PASSWORD
lan set 6 user
lan set 6 password NASZE_TAJNE_HASLO
lan set 6 access on
lan set 6 vlan id off
lan set 6 macaddr MAC:adres:karty:z:IPMI
isol set enabled true
isol set bit-rate 115.2
# u mnie poniższa opcja nie chciała zadziałać, ale da się to obejść
lan set 6 arp respond on
```

Następnie możemy połączyć się zdalnie do IPMI przy pomocy `ipmitool -I lan -H 192.168.0.100 shell` lub do konsoli wystawionej na port szeregowy (w wypadku testowanego sprzętu był to /dev/ttyS1, ponadto wymagał użycia starszego trybu isol zamiast sol) - `ipmitool -I lan -H 192.168.0.100 isol activate`. Testowana płyta (SE7501WV2) posiadała także w biosie opcje przekierowania konsoli na port szeregowy (w odróżnieniu od innej testowanej płyty bez IPMI) ta przekierowywała też gruba więc wystarczyło wystawić jądro i getty. Zobacz w Sieci: Howto setup IPMI under Linux (Debian / Sarge) on the Intel SR2300 Server Chassis (Intel Server Board SE7501WV2)[1876] (uwaga od czasu powstania tamtego artykułu trochę się zmieniło - ipmitool obsługuje isol, inne nazwy modułów).

IPMI możemy wykorzystać także lokalnie - np. do monitoringu temperatury (niekiedy daje lepsze wyniki niż lmsensors czy tez mbmon). Prezentuje przykład prostego skryptu który w przypadku wykrycia zbyt wysokiej temperatury wysyła powiadomnienie i dokonuje (dość brutalnego) wyłączenia maszyny:

```
IPMI_INFO=`ipmitool -I open sdr`
date +%T >> /var/log/temperature_`date +%F`
echo "$IPMI_INFO" | grep Temp >> /var/log/temperature_`date +%F`
echo "$IPMI_INFO" | awk '
	BEGIN {
		ret_code=0
	}
	$1=="CPU" && $3=="Temp" && $5 > 68 {
		if ($5 < 200) {ret_code=12; exit 12} else {ret_code=11}
	}
	END {
		exit ret_code
	}
'
SHUTDOWN=$?
if [ $SHUTDOWN -ge 11  ]; then
	sendEmail -f "root@$HOSTNAME" -t admin@email.adress \
		-u "SHUTDOWN INFO from `hostname`" -m "$IPMI_INFO" \
		-s nasz.serwer.smt.puszczajacy.beze.autoryzacji.i.greylist
	if [ $SHUTDOWN -ge 12  ]; then
		ipmitool -I open power off
	fi
fi
```

Warto także zwrócić uwagę na alternatywną implementację protokołu IPMI - freeipmi[1877].

nie tylko IPMI

IPMI nie jest jedynym interfejsem systememów zarządzania sprzętem. W sererach SUNowskich możemy spotakać także różne *LOMy, poniżej porównanie naistotniejszych funkcji poszczególnych systemów obsługi service procesora:

uruchamianie konsoli	`platform console`	`start /SP/AgentInfo/console`	`start /SP/console`	`console` oraz `consolehistory`
wychodzenie z konsoli (rozłączanie)	`Ctrl+E c .`	`Esc (`	`Esc (`	`## .`
zasilanie	`platform set power state`	`set /SP/SystemInfo/CtrlInfo PowerCtrl=on | gracefulloff | forceoff | reset | BIOSSetup`	`stop -force /SYS`	`poweron | poweroff | reset`
pomoc	`help` oraz dodanie do komendy `-h` lub `--help`	`help [komenda]`	`help [komenda]`	`help [komenda]`
uwagi		konsola często źle znosi reset systemu zlecony z SP (należy ją odpalić ponownie)		

W przypadku ILOM w wersji 1 i 2 możliwe jest zalogowanie się do bash'owej powłoki z użyciem `ssh sunservice@ilom.adres` oraz hasła użytkownika root. Umożliwia to m.in. zrestartowanie ILOMu gdy `restart /SP` odmawia współpracy.

Puppet czyli centralne zarządzanie

Gdy administrujemy wieloma (zwłaszcza podobnymi) systemami uciąliwe staje się wprowadzanie takich samych bądź podobnych zmian w dziesiątkach (jak nie więcej) plików. Dużym ułatwieniem może okazać się (gdy już przebijemy się przez jego konfigurację) system puppet który umożliwia m.in. rozsyłanie plików konfiguracyjnych, generowanie ich w oparciu o szablony, edycję konfiguracji na zdalnym serwerze z użyciem augeas (augtool), instalację oprogramowania i wiele innych.

Typowo na konfigurację składają się:

— pliki `/etc/puppet/manifests/nodes/*.pp` opisujące poszczególne hosty - zawierają sekcje node opisujące danego hosta (definicje specyficznych dla hosta zmiennych, włączanie klas, plików dziedziczenie, …). Przykład:

```
node 'moj.serwer.moja.domena' inherits default {
        # dziedziczymy po node "default"
        $my_ipaddr = '192.168.0.100'
        $my_ipaddr = '192.168.0.101'
        # klasa z parametrami
        class { 'interfaces':
                # numer vlan'u
                vlan => '22',
                # lista adresów ip
                ipaddrs  => [ $my_ipaddr, $my_ipaddr2 ],
                # dodatkowy routing (lista zestawów wartości)
                extra_route => [
                        { 'net' => '192.168.10.0/24',  'gw' => '192.168.0.10' },
                ]
        }
        # włączamy klasę monit
        include monit
}
```

— pliki klas `/etc/puppet/manifests/classes/*.pp` definiujące działania (np. tworzenie pliku konfiguracyjnego z szablonu) Przykład:

```
class apt_source {
        file { '/etc/apt/sources.list':
                owner   => root,
                group   => root,
                mode    => 644,
                content => template('sources.list.${::lsbdistcodename}.erb'),
        }
}
```

— pliki szablonów `/etc/puppet/templates/etc/*.erb` mogą kożystać ze zmiennych - w tym poprzez pętle z list - na przykład:

```
# My vlan is <%= vlan %>
# Routing table is:
<% if ! extra_route.empty? then -%>
<% extra_route.each do |host| -%>
# <%= host['net'] %> via <%= host['gw'] %>
<% end -%>
<% end -%>
<%= scope.function_template(dodatkowy_szablon_do_wstawienia) %<
```

Zobacz też przykładowe moduły puppet'a[1878] w dziale "archiwum"

Monit

Istonym zaganieniem jest kontrola działania usług i podejmowanie akcji gdy coś nie działa. Pomocnym w tym może być monit. Umożliwia on wysyłanie powiadomien, wykonanie restartu niedziałającej usługi, itd.

Dobrym pomysłem wydaje się pozostawienie `/etc/monit/monitrc` bez zmian i umieszczenie całości konfiguracji w `/etc/monit/conf.d/`. Pliki są includowane wg kolejności nazw zatem np. w `/etc/monit/conf.d/000_global.conf` należy umieścić konfigurację podstawową:

```
set daemon 120 with start delay 60
set eventqueue basedir /var/monit slots 100
set logfile syslog facility log_daemon
set mailserver primary.mailserwer, secondary.mailserwer, localhost
set alert adres@mailowy.uzywany.do.powiadomien
set mail-format {
  from: monit@nazwa.monitorowanego.hsta
  subject: [$HOST] monit alert -- $EVENT $SERVICE
}
set httpd port 2812 use address localhost allow localhost
```

W kolejnych plikach mozemy umieścić monitorowanie stanu systemu:

```
check system nazwa.monitorowanego.hsta
  group system
  if loadavg (1min) > 6 then alert
  if loadavg (5min) > 4 then alert
  if memory usage > 95% then alert
  if cpu usage (user)   > 90% then alert
  if cpu usage (system) > 50% then alert
  if cpu usage (wait)   > 35% then alert
```

oraz monitorowanie poszczególnych usług - np.:

```
check process cron with pidfile /var/run/crond.pid
  group system
  start program = "/etc/init.d/cron start"
  stop  program = "/etc/init.d/cron stop"
  if 5 restarts within 5 cycles then timeout
```

Dane o stanie usług prezentowane są na www (w powyższym przykłądzie port 2812). Przy pomocy prostego skryptu można je zebrać w jedn spójny system

MULTIMONIT.PHP

```
<?php


$hosts_file = file('/etc/hosts');
if ($hosts_file === false) {
 echo "Unable to open hosts file.\n";
 exit;
}

$delim = " \t";
$HOSTS = array();

foreach ($hosts_file as $h) {
 if ($pos = strcspn($h, "\r\n#")) {
  $h = trim(substr($h, 0, $pos));
  // mamy usuniete komentarze itp.

  $pos = strcspn($h, "\t ");
  $ip    = substr($h, 0, $pos);
  $name = trim(substr($h, $pos+1));

  // usuniecie dodatkowych nazw
  $pos = strcspn($name, "\t ");
  if ($pos)
   $name    = substr($name, 0, $pos);

  if (preg_match('/-mon$/', $name))
   //$HOSTS["$name"] = $ip;
   $HOSTS["$name"] = $name;
 }
}

// lista może być ustawiona lub uzupełniona w tym pliku wpisami do tablicy $HOSTS postaci:
// $HOSTS["localhost"] = "127.0.0.1";
// $HOSTS["server1"] = "192.168.0.13";


$REQUEST_URI = substr($_SERVER['REQUEST_URI'], 12);
$pos          = strpos($REQUEST_URI, "/");

if ($pos != "") {
 $HOST = substr($REQUEST_URI, 0, $pos);
 $CMD   = substr($REQUEST_URI, $pos+1);
 $pos   = strcspn($CMD, "/?");
 if ($pos != "") {
  $CMD   = substr($CMD, 0, $pos);
 }
} else {
 if($REQUEST_URI == "") {
  echo           "<html><head><title>Monit servers list</title></head><body><h1>Monit servers
list:</h1>
  <ul>";
  while ( list($k, $v) = each($HOSTS) ) {
   echo "<li><a href=\"$k/\">$k</a></li>";
  }
  echo "</ul></body></html>";
 } else {
  header( "Location: " . $_SERVER['SCRIPT_URI'] . "/" ) ;
```

```
   echo "Redirect to: ". $_SERVER['SCRIPT_URI'] . "/";
  }
  exit;
}

$IP = $HOSTS[$HOST];
if ($IP == "") {
  echo "$HOST : Nieznany host";
  exit;
}


$handle = fopen("http://$IP:2813/$CMD", "rb");
if ($handle === false) {
  echo "Unable connect to $IP\n";
  exit;
}
$contents = stream_get_contents($handle);
fclose($handle);

$contents = ereg_replace(
  "Running Monit on more than one server. Use <a[^<>]*>[^<>]*</a> to manage all your Monit ↩
  instances",
  "<a href=\"../\">List All Monit Servers</a>",
  $contents
);

echo $contents;

?>
```

Fail2ban - odpieranie ataków siłowych

Częstym problemem (zwłaszcza w ssh) jest znaczna liczba (często realizowanych przez kiepskie boty) ataków siłowych. Wygodnym narzędziem do zabezpieczenia się przed nimi jest fail2ban. Jego działanie polega na analizowaniu wskazanego logu, wychwytywaniu prób ataku w oparciu o zadane kryteria i dodawaniu do firewala systemowego własnych regułek. W przypadku Linuxa fail2ban dodaje je do własnego łańcucha iptables regułek blokujących (łańcuch ten jest wstawiany na samym początku reguł). Przykładowa konfiguracja[1879]. Podobnym, ale trochę mniej konfigurowalnym (za to wspierającym IPv6) narzędziem jest sshguard[1880].

Logi systemowe - syslog-ng i logrotate

Kuszącą wizją jest zbieranie logów z wszystkich usług poprzez jeden program, który umożliwiłby ich zapisywanie czy też przesyłanie do zdalnej maszyny. Jest to możliwe z wykorzystaniem odpowiedniej konfiguracji usług (problemem mogą być ”młode usługi” jak voip czy xmpp) i np. syslog-ng. Zamieszczam przykładową konfigurację dla syslog-ng oraz stosowny plik konfiguracyjny dla logrotate (należy umieścić go w `/etc/logrotate.d/` i usunąć lub zmodyfikować z tamtąd pliki które obsługiwałyby te same logi)

Konfiguracja większości usług do korzystania z logowania przez syslog zaprezentowana jest wraz z opisami tych usług (głównie w Konfiguracja serwerów sieciowych), poniżej prezentuję uwagi konfiguracyjne dotyczące oprogramowania tam nie opisanego:

- w `postgresql.conf` ustawiamy `log_destination = 'syslog'`
- w `/etc/clamav/clamd.conf` ustawiamy `LogSyslog true`, `LogTime false` i usuwamy `LogFile`
- w `/etc/clamav/freshclam.conf` ustawiamy `LogSyslog true`, `LogTime false` i usuwamy `LogFile` oraz `UpdateLogFile`

SYSLOG-NG.CONF

```
options {
# nie wpisujemy nazwy hosta do logów
chain_hostnames(0);

time_reopen(10);
time_reap(360);
log_fifo_size(2048);

# może tworzyć katalogi
create_dirs(yes);

# prawa do plików logów
owner(root);
group(adm);
perm(0640);
dir_perm(0755);

# nie używamy DNS
use_dns(no);
};

source all_src {
internal();
unix-stream("/dev/log");
file("/proc/kmsg" log_prefix("kernel: "));
};


#######################
#                     #
#   DEFINICJE LOGOW   #
#                     #
#######################


#
# EMERGENCY
#

destination std_console { file("/dev/ttyS0"); file("/dev/tty1"); };
destination all_console { usertty("*"); };
destination emergency_fd { file("/var/log/emergency.log"); };

filter emergency { level(crit,alert,emerg); };

log {
source(all_src);
filter(emergency);
destination(std_console);
destination(all_console);
destination(emergency_fd);
};


#
# poczta - exim, dovecot
#
```

```
destination exim_fd { file("/var/log/exim.log"); };
destination dovecot_fd { file("/var/log/dovecot.log"); };
destination mail_fd { file("/var/log/mail-else.log"); };

filter exim { facility(mail) and program("exim"); };
filter dovecot { facility(mail) and program("dovecot"); };
filter mail { facility(mail); };

log {
source(all_src);
filter(exim);
destination(exim_fd);
flags(final);
};
log {
source(all_src);
filter(dovecot);
destination(dovecot_fd);
flags(final);
};
log {
source(all_src);
filter(mail);
destination(mail_fd);
flags(final);
};


#
# www - apache
#

destination apache_fd { file("/var/log/apache.log"); };
destination www_fd { file("/var/log/www.log"); };

filter apache { facility(daemon) and program("apache"); };
filter www { facility(local0) and program("www"); };

log {
source(all_src);
filter(apache);
destination(apache_fd);
flags(final);
};
log {
source(all_src);
filter(www);
destination(www_fd);
flags(final);
};


#
# systemowe - cron, jądro
#

destination cron_fd { file("/var/log/cron.log"); };
destination kernel_fd { file("/var/log/kernel.log"); };
destination kernel_debug_fd { file("/var/log/kernel-debug.log"); };
```

```
filter kern { facility(kern); };
filter kernel_debug { facility(kern) and level(debug); };
filter cron { facility(cron) or facility(auth) and program("cron"); };

log {
source(all_src);
filter(cron);
destination(cron_fd);
flags(final);
};
log {
source(all_src);
filter(kernel_debug);
destination(kernel_debug_fd);
flags(final);
};
log {
source(all_src);
filter(kern);
destination(kernel_fd);
flags(final);
};


#
# autoryzacja
#

destination auth_fd { file("/var/log/auth.log"); };
destination auth-err_fd { file("/var/log/auth.err.log"); };

filter auth { facility(auth); };
filter auth-err { facility(auth) and level(warning,err,crit,alert,emerg); };

log {
source(all_src);
filter(auth-err);
destination(auth-err_fd);
flags(final);
};
log {
source(all_src);
filter(auth);
destination(auth_fd);
flags(final);
};


#
# bazy danych - mysql, postgres
#
destination mysql_fd { file("/var/log/mysql.log"); };
destination postgres_fd { file("/var/log/postgres.log"); };

filter mysql { program(".*mysql.*"); };
filter postgres { program("postgres"); };

log {
source(all_src);
filter(mysql);
```

```
destination(mysql_fd);
flags(final);
};
log {
source(all_src);
filter(postgres);
destination(postgres_fd);
flags(final);
};


#
# wszystko inne (co sięnie załapało wcześniej)
#

destination elselog_fd { file("/var/log/elselog.log"); };

log {
source(all_src);
destination(elselog_fd);
flags(final);
};
```

SYSLOG-NG

```
/var/log/dovecot.log /var/log/mail-else.log /var/log/apache.log /var/log/www.log ⤸
/var/log/cron.log /var/log/kernel.log /var/log/auth.log /var/log/mysql.log /var/log/postgres.⤸
log /var/log/elselog.log {
   rotate 4
   missingok
   notifempty
   weekly
   compress
   postrotate
      /etc/init.d/syslog-ng reload >/dev/null
   endscript
}

/var/log/emergency.log /var/log/auth.err.log {
   rotate 6
   missingok
   notifempty
   weekly
   compress
   postrotate
      /etc/init.d/syslog-ng reload >/dev/null
   endscript
}

/var/log/exim.log /var/log/kernel-debug.log {
   rotate 12
   missingok
   notifempty
   weekly
   compress
   postrotate
      /etc/init.d/syslog-ng reload >/dev/null
   endscript
```

```
}
```

Usługi wysokiej dostępności

Nierzadko mamy doczynienia z koniecznością zapewnienia (niemal) bezprzerwowej dostępności jakiejś usługi (np. serwera www). Z pomocą przychodzi nam tutaj oprogramowanie takie jak heartbeat[1881], czy też nawet cały openais[1882] wraz z innymi narzędziami takimi jak ctdb[1883], drbd8-utils[1884] (klastrowa baza danych tymczasowych oraz raid1 over tcp/ip).

Zasada działania heartbeat[1885] polega na kontrolowaniu działania hosta (poprzez komunikację z swoim klientem) i w przypadku wykrycia padu tamtego hosta automatycznym uruchomieniu odpowiednich usług (i przeniesieniu adresu) na inny host.

Poradnik *BSD

Ogólna konfiguracja

Zgodnie z unixową tradycją informacje konfiguracyjne przechowywane są w plikach umieszczonych w katalogu `/etc`. Do najważniejszych należy zaliczyć:

- `/etc/sysctl.conf` - konfiguracja parametrów jądra
- `/etc/rc.*` - zarządzanie startem systemu, w szczególności:
  - `/etc/rc.d/XXX.sh`, `/usr/local/etc/rc.d/XXX.sh` - skrypty startowe (FreeBSD), za ich aktywację odpowiadają wpisy `XXX_enable` w `/etc/rc.conf`
  - `/etc/rc.conf` - w OpenBSD pełnia konfiguracji systemu (dość długi skrypt), w FreeBSD oprócz wspomnianego wyboru skryptów startowych także parametry startu takie jak konfiguracja sieci.
- `/etc/defaults/*` - w FreeBSD parametry dodatkowe skryptów startowych

Mamy możliwość manualnej konfiguracji sieci poprzez komendy typu ifconfig, route itp, np.:

```
# dodatkowy adres IP:
ifconfig bge0 inet 192.168.24.13 netmask 255.255.0.0 alias
# w FreeBSD zamiast alias można dać tez add
# ustawienie routingu domyślnego
route add default 192.168.0.1
# informacja o routingu (odpowiednik linuxowego route -n /ip route sh):
netstat -rn
```

Możliwe jest też bardziej wyrafinowane (niż wrzucenie tych komend do skryptu startowego) skonfigurowanie sieci na stałe, np. w FreeBSD służą do tego opcje w `/etc/rc.conf`:

```
ifconfig_bge0="inet 192.168.24.1 netmask 255.255.0.0"
ifconfig_bge0_alias0="inet 192.168.24.13 netmask 255.255.0.0"
defaultrouter="192.168.0.1"
# lub dla DHCP na domyślnym interfejsie:
ifconfig_DEFAULT="DHCP"
```

Systemy z rodziny *BSD oferują jeden lub kilka (FreeBSD wszystkie trzy) z następujących mechanizmów firewall'a: ipf[1886], ipfw[1887], pf[1888]. Zazwyczaj za realizację funkcji typu NAT, jak i bardziej zaawansowanego routingu odpowiadają osobne daemony. Warto zainteresować się także programem netfwd[1889].

W FreeBSD dość istotne parametry można także ustawiać za pomocą `sysinstall`. W tym miejscu warto zwrócić uwagę drobne różnice składniowe w standardowych komendach - np. ifconfig.

Z innych przydatnych administracyjnych komend FreeBSD należy wspomnieć o:

- `systat` - różne informacje o systemie (np. z opcją `-tcp` połączenia TCP)
- `sysctl` - różne informacje o systemie i ustawienia (odpowiednik systemu `/proc`, np. `sysctl hw` wyświetli informacje o sprzęcie)
- `truss` - śledzenie wywołań systemowych programu
- `pciconf` - zarządzenie (w tym listowanie urządzeń) PCI

— `usbdevs` - zarządzenie (w tym listowanie urządzeń) USC
— `atacontrol` - zarządzenie (w tym listowanie urządzeń) ATA
— `camcontrol` - zarządzenie (w tym listowanie urządzeń) SCSI

Instalacja oprogramowania

Zarówno FreeBSD, jak i OpenBSD oferują oprogramowanie w postaci pakietów binarnych, jak i systemu portów. Porty to przygotowane do łatwego kompilowania źródła (coś na kształt pakietów źródłowych). Instalacja programów znajdujących się w paczkach binarnych jest dość prosta i sprowadza się do wydania komendy `pkg_add -r nazwa_pakietu`. Aby to zadziałało oczywiście pakiet musi znajdować się w repozytorium w postaci pliku `nazwa_pakietu.tbz`. Natomiast sam adres repozytorium może być modyfikowany poprzez ustawianie zmiennej środowiskowej `PACKAGESITE` np. `export PACKAGESITE=ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-6.4-release/Latest/`. Wyeksportowanie tej zmiennej może być niekiedy konieczne do instalacji pakietu.

Znalezienie w jakim pakiecie jest potrzebna nam aplikacja ułatwiają serwisy takie jak: http://www.freebsd.org/ports/[1890]. Do zarządzania pakietami przydają się także komendy takie jak: `pkg_info`, `pkg_version` i `pkg_delete`.

Dyski

Systemy *BSD korzystają z własnego sytemu partycjonowania dysku (tzw. disklabel). Polega to na tym iż na wybranym dysku (`/dev/adX`, w OpenBSD `/dev/wd...`) w ramach jednej "zwyczajnej" partycji (zwanej partycją fdiskową, lub segmentem) `/dev/adXsY` tworzone są partycje BSD (oznaczane literami od a do h). Więcej na temat tego mechanizmu w Tworzenie partycji z wykorzystaniem Disklabel[1891].

W tym miejscu warto zwrócić uwagę na podobieństwo w zarządzaniu dyskami do Solarisa. Standardowymi dla tych systemów systemami plików są FFS[1892], UFS[1893].

Debian GNU/kfreebsd

Instalacja

Proces instalacji systemu z rootfs na zfs opisałem szczególowo na protalu DUG[1894]. Najważniejsze wydaje się zastosowanie tablicy partycji GPT i wgranie w nią GRUB2 o czym piszę także tutaj.

Konsola szeregowa

Aby wystawić konsolę Debian GNU/kfreebsd na pierwszy port szeregowy z prędkością 9600 należy:

— wpisać konsole na cuau0 do `/etc/inittab` - `T0:23: respawn:/sbin/getty -L cuau0 9600 vt100`
— dodać ją do `/etc/securetty`
— (aby ogladać komunikaty startowe) w `/etc/grub.d/10_kfreebsd` w linii wypisujacej "kfreebsd" (`kfreebsd ${rel_dirname}/${basename}`) dopisać na koncu `-D -h`
— wydać polecenie `update-grub2`

Pełniejszy zestaw opcji w command line gruba po wpisaniu `kfreebsd --help`

Więcej informacji

Zobacz w Sieci: Podręcznik FreeBSD[1895], OpenBSD - Dokumentacja i FAQ[1896], Wprowadzenie do FreeBSD i Uniksa[1897], Mosty filtrujące[1898].

Poradnik *Solarisoy

Komendy do zarządzania systemem

- `pfexec` - wykonanie komendy uprzywilejowanej przez użytkownika, którego rola na to pozwala
- `svcs`, `svcadm` - konfiguracja uruchamiania usług podczas startu systemu
- `pkgadd -d $FILEMANE` - instalacja paczek z oprogramowaniem z poza repozytoriów (`pkgrm `pkginfo -d $FILEMANE | cut -f2 -d' '`` - usuwanie tak zainstalowanych paczek)
- pkg - zarządzanie oprogramowaniem w oparciu o repozytoria (np. `pkg search -r vim` - wyszukanie w jakich paczkach nielokalnych jest plik vim, `pkg install SUNWvim SUNWscreen SUNWtop` - instalacja vim'a screen'a i top'a)
- `crle -l /lib:/usr/lib:/usr/local/ssl/lib` - konfiguracja ścieżek wyszukiwania bibliotek - odpowiednik ldconf z Linuxów (`LD_LIBRARY_PATH` współcześnie nie używana)
- `devfsadm` - zarządzanie urządzeniami w `/dev/`
- `netstat` - statystyki sieci (z opcją `-r` pokazuje tablice routingu, z opcją `-s` pokazuje statystyki dla protokołów, z opcją `-i` pokazuje statystyki ruchu IP dla interfejsów)
- `prstat` - statystyki aktywnych procesów
- `mpstat` - statystyki dla procesów i grup procesów
- `sar` - raporty aktywności systemu
- `vmstat` - statystyki pamięci wirtualnej
- `swap` - administracja oraz pokazanie wykorzystania przestrzeni swap
- `zoneadm` - administracja zonami (odpowiednik jail'i)

Sieć

Konfiguracja sieci odbywa się w kilku plikach konfiguracyjnych - są to:

- `/etc/hostname.INTERFACE` - określenie nazwy hosta dla zadanego przez INTERFACE interfejsu sieciowego, kilka nazw na interfejsie uzyskuje się poprzez stosowanie aliasów INTERFACE:X, gdzie X jest kolejnym numerem
- `/etc/hosts` - skojarzenie nazw hostów z numerami IP (w szczególności nazw przypisanych do interfejsów z numerami IP które będą do nich przypisane)
- `/etc/netmasks` - przypisania masek do adresów sieci IP
- `/etc/defaultrouter` - domyślna trasa routingu (brama)
- `/etc/nodename` - nazwa hosta
- DNS konfigurujemy poprzez wpisy w dwóch plikach - "`hosts: files dns`" w `/etc/nsswitch.conf` i typowy konfig w `/etc/resolv.conf`

Sieć opartą na takiej konfiguracji uruchamiamy `svcadm enable svc:/network/physical:default` (przed tym należy wyłączyć inne sposoby konfiguracji sieci - w szczególności `svcadm disable svc:/network/physical:nwam`).

Użytkownicy

Konta użytkowników konfiguruje się w standardowych `/etc/passwd` `/etc/shadow` `/etc/group` (które możemy zarządzać bezpośrednio bądź poprzez komendy typu `useradd`). Natomiast system ten posiada mechanizm "ról" pozwalający na nadanie pewnym użytkownikom większych praw (coś na kształt sudo) - mechanizm ten konfigurowany jest w pliku `/etc/user_attr`, który możemy zarządzać także przy pomocy komendy `roleadd` i pokrewnych.

Konsola na porcie szeregowym

Jest to szczególnie przydatne w Solarisach działających na maszynach z jakimś ILOM/ALOM/ELOM. Aby uruchomić ją należy zadbać o następujące wpisy (zakładamy tutaj iż interesującym nas portem jest pierwszy port szeregowy - może się to różnić w zależności od konfiguracji sprzętowej) w `/boot/solaris/bootenv.rc`:

```
setprop ttya-rts-dtr-off false # na SUN FIRE X4540 true
setprop ttya-ignore-cd true
```

```
setprop ttya-mode 9600,8,n,1,-
setprop console text
```

oraz w `/rpool/boot/grub/menu.lst` (ta ścieżka wskazuje położenie konfiguracji gruba gdy / jest na zfs'ie, a config dla takiego przypadku jest podany):

```
serial --unit=0 --speed=9600 --ord=8 --parity=no --stop=1
terminal --timeout=5 serial console
timeout 5
default 0
title OpenSolaris
# uwaga: po zbootowaniu z ta konsola nie będzie działała graficzna konsola na ILOM
findroot (pool_rpool,0,a)
bootfs rpool/ROOT/opensolaris
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS,console=ttya -kv -m verbose
module$ /platform/i86pc/$ISADIR/boot_archive
```

Dyski i filesystemy - ZFS, NFS, ...

Na koniec kilka słów na temat zarządzania dyskami oraz ZFS. Urządzenia reprezentujące dyski znajdują się w `/dev/dsk/` i mają nazwy oparte na schemacie `c{numer_kontrolera}t{numer_"tasku"}d{numer_dysku_w_tasku}`, na końcu może być `sX` oznaczające partycje solarisowa X (numerujemy od zera) lub `pX` oznaczający odpowiednią partycje x86 (0 == cały dysk). `pX` nie występują na sparc'ach, `s2` odnosi się tradycyjnie do partycji "backup" obejmującej cały dysk (partycje solarisowe mogą się nakładać). Do zarządzania dyskami służy komenda `format`, w jej ramach mamy do dyspozycji polecenia takie jak:

- `fdisk` - partycje x86 (fdisk możemy też wywołać niezależnie korzystając do tego celu z urządzenia `/dev/rdsk/c0t0d0p0`
- `verify` - wyświetlanie informacji o podziale fdiskowej (x86) partycji solarisowej na "podpatycje" (slice)
- `partition` - partycje w ramach partycji fdiskowej (x86) lub całego dysku (sparc)
- `disk` - wybór dysku
- `current` - informacje o dysku
- `type` - przełączanie pomiędzy aktywnymi tablicami partycji solarisowych / partycjami x86 na których są partycje o typie solarisowym

Takie podejście do zarządzania dyskami narzuca skojarzenia z OpenBSD i jego "disklabel", jest ono zresztą rzeczą dość charakterystyczną dla nie PC centrycznych systemów.

Do zarządzania i obsługi ZFS służą zasadniczo dwa narzędzia:

- `zpoll` - dyski, ich połączenia (mirrory, raidy, grupy dysków wchodzące w skład danego fs, itp) oraz podstawowy system plików na urządzeniu (np. przy pomocy tej komendy możemy zamienić pool mający jeden dysk na poll oparty o mirror - wystarczy zrobić "attach" - mirror utworzy się samoczynnie)
- `zfs` - dodatkowe ustawienia systemu plików i dodatkowe systemy plików tworzone na założonym wcześniej pool-fs

Udostępnianie zasobów po NFS odbywa się za pomocą komendy `share -F nfs -o rw=hostA:hostB,ro=hostC/sciezka/do/eksportu`. Komendy te mogą być wpisane do pliku `/etc/dfs/dfstab` celem automatycznego ich wykonywania przy starcie systemu. DNS dla hostów podawanych w komendzie share musi być wpisany w `/etc/hosts` (nawet jeżeli jest skonfigurowany normalny DNS). Można także zdefiniować eksport NFSowy w samym ZFSie (jest to o tyle lepsze że nie rodzi problemów z kolejnością podmontowywania i robienia exportu) - służy do tego komenda `zfs set sharenfs=on eksportowany/katalog;`, opcje eksportowania ustawia się wtedy w sposób następujący `sharemgr set -P nfs -S sys -p rw=hostname zfs/eksportowany/katalog`, a sprawdzić można je za pomocą `sharemgr show -p`. Ponadto należy zwrócić uwagę na niską wydajność NFS'a ciągniętego z ZFSa w niektórych zastosowaniach (związaną z realizacją sync) - wybitnym tego przykładem może być konwersja z bmp do tif przy pomocy convert (operacja gdy zapis jest robiony na nfs ta trwa 80 razy dłużej niż gdy jest robiony lokalnie a wynik jest przenoszony na nfs).

Należy też wspomnieć iż istnieje konkurencyjna wobec ZFS technologia tego typu systemów plików, nie mającą obecnie wolnej implementacji, jest nią WAFL[1899].

Zasoby systemowe

W Linuxie większości ustawień systemowych (np. wielkość segmentu pamięci dzielonej) dokonuje się przez system /proc (niekiedy przez /sys). W przypadku Solarisów tradycyjną metodą są ustawienia w /etc/system np.

```
forceload: sys/shmsys
set shmsys:shminfo_shmmax=3221225472
```

A bardziej nowoczesną (i preferowaną) metodą konfiguracja poprzez projekty (szczegóły w man resource_controls). Baza projektów znajduje się w pliku /etc/project, a ustawień dokonuje się poprzez projadd i projmod np.:

```
# utworzenie projektu
projadd user.postgres
# ustawienie parametrów projektu
projmod -s -K "project.max-shm-memory=(priv,8G,deny)" user.postgres
# dodanie użytkownika do projektu
projmod -a -U postgres user.postgres
```

Sztuczki

Często na Solarisie w vim w trybie edycji niepoprawnie działają strzałki - aby to naprawić należy do ~/.vimrc dodać:

```
:map OD <Right>
:map OC <Left>
:map OB <Down>
:map OA <Up>
```

Może się jednak zdarzyć że gdy vim uruchamiamy jako vi te kombinacje nie będą działać wtedy do ~/.bashrc należy dodać:

```
VIM=`which vim`
if [ -x $VIM ]; then
        alias 'vi'="$VIM"
fi
```

Jeżeli korzystamy z tcsh (lub csh) należy odpowiedni konfig przygotować dla tej powłoki, chyba że wolimy basha (a w passwd musi być tcsh bo np. na części maszyn basha nie ma) to możemy wpisać do ~/.login:

```
env | grep DISPLAY > /dev/null
if ( $? != 0 && -x /bin/bash ) then
        exec /bin/bash $argv
endif
```

Solarisowy **cron** wymaga aby użytkownik który z niego korzysta miał w /etc/shadow w polu hasła coś innego niż *LK*, logi crona można znaleźć w /var/cron/log.

Linki

Zachęcam także do zapoznania się z Solaris x86 FAQ[1900].

Multimedia

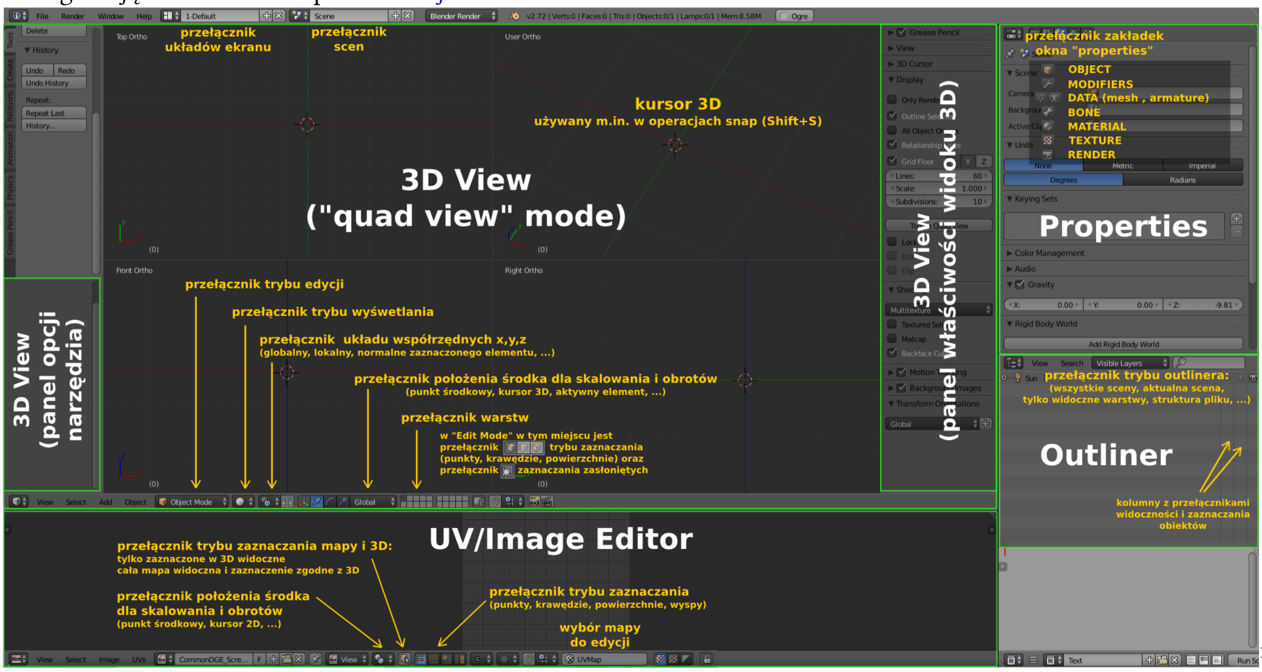
W tym artykule postaram się omówić najważniejsze kwestie związane z odtwarzaniem, przetwarzaniem oraz edycją różnego typu danych multimedialnych (grafika, dźwięk i wideo).

Blender jako narzędzie do tworzenia grafiki dla Ogre

Blender[1901] jest programem do tworzenia i renderowania grafiki oraz animacji 3D. Poniżej zamieszczam kilka przydatnych informacji i porad związanych z jego obsługą nastawionych głównie na wykorzystanie go do tworzenia modeli dla Ogre[1902].

Wprowadzenie

Przykładowy układ okien edycyjnych przedstawiony (i opisany jest na poniżej ilustracji). Plik z taką konfiguracją widoku można pobrać tutaj[1903].



[1904]

Na początek kilka uwag ogólnych:

- blender nie pyta o zapisanie pliku przy wyjściu, ale nie zostawia plik `/tmp/quit.blender`
- blender oczekuje potwierdzenia zapisania przy pomocy skrótu klawiaturowego Ctrl+S
- blender zapisuje układ okien edycyjnych w edytowanym pliku
- blender (w przeciwieństwie do Ogre) standardowo pokazuje wszystkie płaszczyzny dwustronnie, można to wyłączyć (co jest przydatne do korygowania normalnych) - w tym celu należy w panelu właściwości okna "3D View" w sekcji "Shading" zaznaczyć "Backface Culling"
- warto korzystać z przełączników trybu zaznaczania oraz przełącznika położenia punktu odniesienia dla skalowania i obrotów

Podstawy obsługi programu

- obsługa widoku 3D:
  - obracanie: [środkowy przycisk myszy]
  - przesuwanie: [Shift]+[środkowy przycisk myszy]
  - przybliżanie: [kółko przycisk myszy]
  - ustawienie punktu na który patrzy i względem którego obracamy/przybliżamy kamerę na aktualnie zaznaczony: [numpad .] (lub [View] -> [Align View] -> [View Selected])
  - przełączanie "Edit Mode" i "Object Mode": [Tab]
  - ustawianie kursora 3d na zaznaczeniu, zaznaczenia na kursorze 3d itd: [Sfift]+[s]
- zaznaczanie:
  - zaznaczanie/odznaczanie wszystkiego: [a]
  - zaznaczanie ramką: [b], następnie [lewy przycisk myszy] (odznaczanie [Shift]+[lewy przycisk myszy])

- zaznaczanie okręgiem przyczepionym do kursora: [c], następnie [lewy przycisk myszy] (odznaczanie [Shift]+[lewy przycisk myszy])
- zaznaczanie kliknięciem: [prawy przycisk myszy]
- zaznaczanie wielu obiektów kliknięciem (przełączanie zaznaczenia): [Shift]+[prawy przycisk myszy]
-
-
- do zaznaczania możemy także:
  - pozycji w oknie "Outliner" (w trybie edycji obiektów)
  - grup wierzchołków ("Vertex Groups") w zakładce "Data" okna "Properties" (w trybie edycji siatki)

- ukrywanie, pokazywanie ukrytych (pozycja [Hide/Show] w menu [mesh] lub [object]):
  - ukrywanie zaznaczonych wierzchołków: [h]
  - ukrywanie nie zaznaczonych wierzchołków: [Shift]+[h]
  - pokazywanie ukrytych: [Alt]+[h]
- manipulowanie zaznaczonymi obiektami/fragmentami mesh-a:
  - kopiowanie (powielanie):
    - pełne: [Shift]+[d]
    - linkowanie (niedostępne w "Edit Mode"): [Alt]+[d]
  - usuwanie: [Delete]
  - przenoszenie: [g]
  - skalowanie: [s]
  - obracanie: [r]
  - po wyborze przenoszenia/skalowania/obracania:
    - można wskazać kierunek: [x], [y], [z] lub płaszczyznę: [Shift]+[x], [Shift]+[y], [Shift]+[z]
    - podać wartość numeryczną z klawiatury
    - ustalić wartość numeryczną myszą - dostępne są modyfikatory powolnie [Shift] lub z krokiem [ctrl]

Podstawy edycji siatki

- wyciąganie wierzchołków wraz z krawędziami (wytłaczanie): [e]
  podstawowe opcje:
  - only edges - wyciągane będą krawędzie (a między nimi utworzone zostaną ściany)
  - region - wyciągnięty będzie obszar (zewnętrzne ściany), może służyć do robienia wgłębień
  - indyvidual faces (wyciągnięte będą ścianki (wierzchołki gdzie ścianki stykają się pod kątem zostaną powielone))
- wypełnianie krawędzią lub płaszczyzną: [f]
  - 2 wierzchołki => krawędź
  - 3-4 wierzchołki => płaszczyzna z ew. uzupełnieniem krawędzi
  - więcej wierzchołków => płaszczyzny pomiędzy zaznaczonymi krawędziami
- podział:
  - zaznaczonych krawędzi (krawędzie dzielone wg równych proporcji, w panelu opcji narzędzia można podać m.in. liczbę cięć): [Mesh] -> [Normals] -> [Subdive]
  - nożem (knife): [k]
    opcja wymaga narysowania łamanej wg której będzie przeprowadzany podział (odcinki zatwierdza się klikając - punkty przecięć zmienią kolor z zielonych na czerwone) i zatwierdzenia całości Enterem
- redukowanie wierzhołków:
  - łączenie zaznaczonych wierzchołków: [Alt]+[m]
  - usuwanie powielonych wierzchołków (w panelu opcji narzędzia można wybrać odległość determinującą to czy wierzchołki są zdublowane czy nie): [Mesh] -> [Vertices] -> [Remove Doubles]
  - redukcja złożoności mesha: w zakładce "Modifiers" w oknie "Properties" dodajemy modyfikator "Decimate" (fajne efekty daje "planar" około 7st)
- wektory normalne ([Mesh] -> [Normals]):
  - automatyczne przeliczanie normalnych (na ogół działa dobrze, w panelu opcji narzędzia można wybrać czy ma ustawiać na zewnątrz czy do wewnatrz): [Ctrl]+n
  - odwrócenie normalnych:
- nadawanie grubości: w zakładce "Modifiers" w oknie "Properties" dodajemy modyfikator "Solidfy"
- wydzielanie zaznaczenia do osobnego obiektu: [p] -> [Selection]

Grupy wierzchołków

Blender umożliwia grupowanie wierzchołków wewnątrz pojedynczego mesha w ramach ”Vertex Group”. Ma to zastosowanie m.in. do przypisywania kości (jest też to bardzo przydatne ze względu na to że eksport do Ogre odbywa się na zasadzie jeden mesh w Blenderze = jeden mesh w Ogre).

1. w ”Edit Mode”:
   a) zaznaczenie wierzchołków które chcemy przypisać do grupy
   Mesh -> [Vertices] -> [Vertex Groups] i:
      Assign to New Group aby utworzyć nową grupę i przypisać do niej
      Assign to Active Group aby utworzyć przypisać ”aktywnej grupy” (ustawia się ją poprzez [Set Active Group] w tym samym menu)
2. nazwy grup możemy zmienić w zakładce ”Data” okna ”Properties” lub w oknie ”Outliner”

Kilka materiałów w jednym meshu

Jeżeli chcemy nałożyć na fragmenty obiektu inny mteriał niż na całą resztę należy (po przypisaniu materiału podstawowego):

1. w ”Edit Mode” zaznaczyć powierzchnie (wierzchołki) na które chcemy nałożyć inny materiał
2. w zakładce ”Material” okna ”Properties” wytbrać/utworzyć materiał który chcemy dla nich zastosować i przypisać go do nich klikając ”Assign”

Nie należy szaleć z ilością materiałów zastosowanych w ramach pojedynczego mesh-a gdyż Ogre ma limitowaną ich ilość.

Teksturowanie

Jeżeli chcemy nałożyć na obiekt (np. ścianę) teksturę która będzie powtarzana i nie skalowana wraz z obiektem:

1. w oknie ”Properties”:
   a) utworzenie materiału w zakładce ”Material”
   b) utworzenie związanej z tym materiałem tekstury w zakładce ”Texture”
   c) ustawienie typu tej tekstury na image i wybór odpowiedniego obrazka
   d) ustawiamy dla niej ”Coordinates” w ”Mapping” na ”Global”
   e) przy pomocy ”size” dostosowujemy wielkość tekstury w stosunku do obiektu

Teksturowanie UV

Jeżeli chcemy nałożyć na obiekt teksturę która będzie na nim mapowana wg określonych współrzędnych:

1. w ”Edit Mode”:
   a) zaznaczenie krawędzi, w których chemy rozciąć siatkę celem zmapowania na płaską teksturę i oznaczenie ich jako szwów poprzez [Mesh] -> [Edges] -> [Mark Seam]
   b) zaznaczenie wierzchołków których mają być mapowane (nie musimy mapować na teksturę całości obiektu) oraz wykonanie ich mapowania poprzez [Mesh] -> [UV Unwraping] -> [Unwrap] (możemy wybrać inny tryb mapowania)
2. w oknie ”UV/Image Editor”:
   a) wczytanie tekstury
   b) dostosowanie rozkładu wierzchołków w oknie UV (przydatne jest ukrywanie fragmentów mesha)
3. w oknie ”Properties”:
   a) utworzenie materiału w zakładce ”Material”
   b) utworzenie związanej z tym materiałem tekstury w zakładce ”Texture”
   c) ustawienie typu tej tekstury na image i wybór odpowiedniego obrazka
   d) ustawienie typu mapowania tej tekstury na UV i wybór odpowiedniej mapy

Tworzenie tekstur i map normalnych

Za przykład pliku z przygotowanymi obiektami do takiej operacji może posłużyć: Creating seamless Textures of diffrent Surfaces[1905]

1. w oknie ”3D View”:
   a) zaznaczamy trójwymiarowy obiekt
   b) zaznaczamy z shiftem płaszczyznę nad nim
   c) obiekt będzie oznaczony na pomarańczowo, a płaszczyzna na żółto)
2. w oknie ”Properties” w zakładce ”Render”:
   a) otwieramy sekcję ”Bake”
   b) ustawiamy tryb na ”Normals” i przestrzeń ”Tangent”
   c) naciskamy przycisk ”Bake”
3. w oknie ”UV/Image Editor”:
   a) powinniśmy zobaczyć wygenerowaną teksturę, a przy menu ”Image” powinna pojawić się gwiazdka
   b) w menu ”Image” wybieramy ”Save as Image”
4. ponawiamy kroki 2 i 3 wybierając tryb ”Texture” (jeżeli z jakiś powodów potrzebujemy odzwierciedlić w teksturze efekt krzywizn możemy użyć ”Ambient Occlusion” lub ”Full Render”)

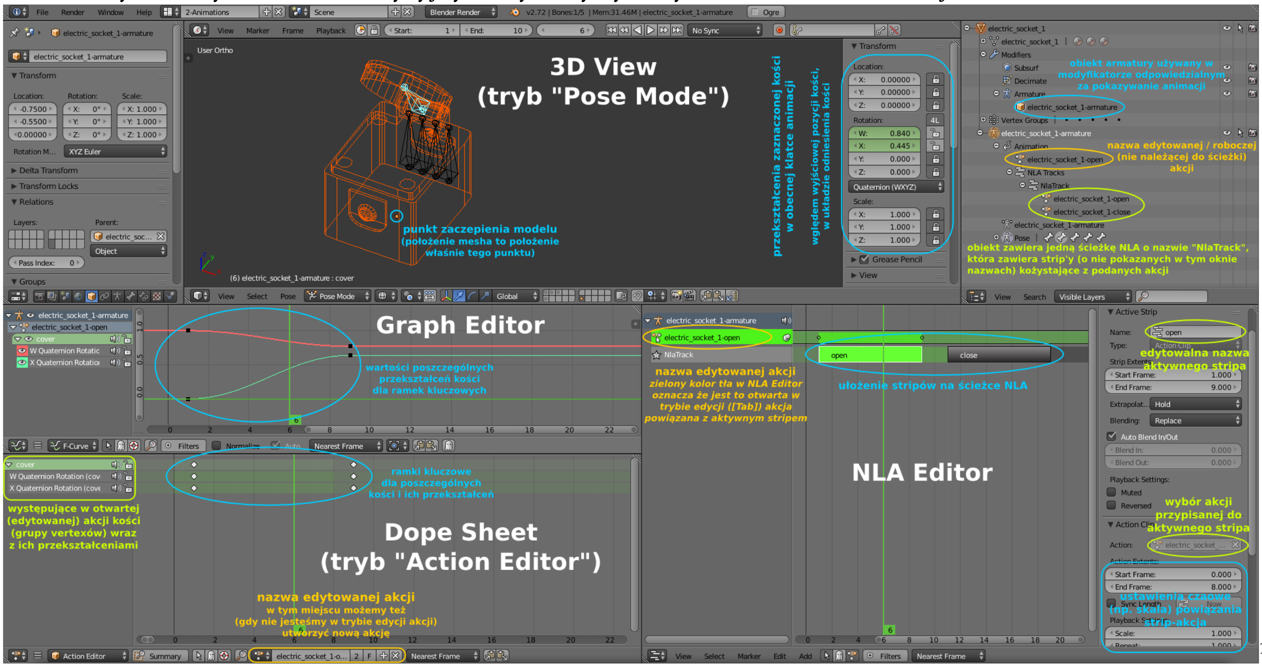
Uzyskaliśmy dwa pliki tekstur - zwykłej i mapy normalnych dla wybranego obiektu. Technikę tę możemy stosować nie tylko dla płaszczyzn ale także dla bardziej skomplikowanych obiektów - więcej informacji[1906].

Dodawanie kości

1. w ”Object Mode” dodajemy armaturę [Add] -> [Armature] -> [Single bone]
2. w oknie ”Properties” w zakładce ”Object” w sekcji ”Relations” ustawiamy jako ”Parent” utworzonej armatury obiekt z który ma obsługiwać
3. mając zaznaczoną uzyskaną kość przechodzimy do ”Edit Mode”
4. dodajemy kolejne kości
   — Ogre wymaga aby statyczne wierzchołki także były skojarzone z jakąś kością
   — warto dla każdej z grupy wierzchołków zdefiniować osobną kość (umożliwia to m.in. zachowanie informacji o grupach wierzchołków przy eksporcie)
5. w oknie ”Outliner” ustawiamy odpowiednie nazwy naszym kościom -
6. mając zaznaczony obiekt do którego przynależna jest armatura w oknie ”Properties” w zakładce ”Modifiers” dodajemy modyfikator ”Armature” i ustawiamy w nim ”Object” na utworzoną wcześniej armaturę

Tworzenie animacji

Przykładowy układ okien edycyjnych wykorzystywanych w tworzeniu animacji:



[1907]

1. w oknie "3D View" zaznaczamy armaturę animowanego obiektu i wybieramy tryb "Pose Mode"
2. w oknie "Dope Sheet" w trybie "Action Editor" dodajemy nową akcję i nadajemy jej unikalną nazwę (np. na zasadzie nazwa_obiektu-nazwa_akcji)
3. przygotowujemy klatki kluczowe animacji:
   a) w "Action Editor" przesuwając zielony wskaźnik ustawiamy numer klatki animacji
   b) w "Pose Mode" ustawiamy kości a następnie zapisujemy ramkę poprzez [Pose] -> [Animation] -> [Insert Key Frame...] (alternatywnie [i]) i wybranie co ma zostać zapisane (np. whole character)
   c) ponawiamy dwa poprzednie kroki dla kolejnych ramek
   d) manipulując markerami klatek kluczowych dla poszczególnych atrybutów poszczególnych kości w "Action Editor" możemy dokonywać korekt przygotowywanej animacji
4. w oknie "NLA Editor":
   a) w lewym panelu klikamy podwójną strzałkę w dół przy utworzonej akcji aby dołączyć animację do ścieżki
   b) w głównej sekcji naciskamy [g] i umieszczamy utworzony tak aby nie kolidował z innymi dla danego obiektu (bez względu czy są czy nie na tej samej ścieżce)
   c) w prawym panelu w sekcji "Active Strip" ustawiamy (nieunikalną) nazwę animacji, której będziemy używać w Ogre
   d) w prawym panelu w sekcji "Action Clip" możemy zmienić animację podłączoną do aktualnego , użyć tylko fragmentu animacji, zmienić jej prędkość
   e) zaznaczone i ścieżki możemy usuwać przy pomocy [x]
5. wracamy do kroku 2 dla kolejnej animacji (w "Action Editor" możemy kopiować i np. odbijać w czasie wcześniej przygotowane animacje, co jest przydatne przy tworzeniu animacji zamknij/otwórz)

Eksport do Ogre

1. Należy pobrać i zainstalować odpowiednią do posiadanej wersji blendera wtyczkę "blender2ogre" - http://code.google.com/p/blender2ogre/[1908], https://bitbucket.org/cc8192/blender2ogre/src[1909] (plik io_export_ogreDotScene.py)
2. Warto rozsądnie nazwać eksportowane mesh'e i materiały.
3. Warto jest stosować manualnie przygotowaną biblioteczkę materiałów w formacie Ogre, a w Blenderze używać niezależnie im podobnych o takich samych nazwach. Eksportery słabo sobie radzą z materiałami.
4. Obiekty które chcemy mieć w postaci pojedynczego mesh'a należy scalić ([Object] -> [Join]) do postaci pojedynczego mesha w blenderze. Przed scaleniem należy:
   a) w przypadku obiektów linkowanych należy je rozlinkować - w tym celu po ich zaznaczeniu należy wybrać [Object] -> [Make Single User] -> [Object & Data]
   b) zaaplikować modyfikatory poprzez "Apply" w zakładce "Modifiers" w oknie "Properties"

c) zaaplikować skalowanie [Object] -> [Apply] -> [Scale]
5. Należy zaznaczyć obiekty do wyeksportowania i wybrać [File] -> [Export] -> [Ogre 3D (.scene and .mesh)]
6. Jeżeli chcemy móc składać animacje (np obrót ramienia i opuszczanie z niego liny) to należy zaznaczyć "Only Animated Bones"
7. W uzyskanym pliku .scene należy dodać sekcje <environment> i ew. <terrain>
8. Jeżeli materiał jest całkiem czarny należy w jego konfiguracji skorygować wartość ambient (aby była nie zerowa)

Alternatywnie do eksportu mesh-y można użyć https://code.google.com/p/torchlight-to-blender/[1910]. Jako że osobne mesh'e blenderowe eksportuje jako submesh'e Ogre bez uwzględniania przesunięć ustawionych w trybie obiektowym, przy jego użyciu nie trzeba łączyć obiektów blenderowych, ale trzeba zaaplikować także ich obroty ([Object] -> [Apply] -> [Rotation]) i położenia ([Object] -> [Apply] -> [Location]). Ten eksporter także (chyba jeszcze większe) ma problemy z eksportem materiałów.

Import z Ogre

Import z Ogre do Blendera umożliwia https://code.google.com/p/torchlight-to-blender/[1911] (nie importuje animacji, ale kości i grupy vertexów tak). Przy imporcie (tak jak i przy eksporcie) ma problemy z materiałami.

Linki

Zobacz w Sieci: Twoja pierwsza animacja w 30 plus 30 minut część I[1912], Podstawy modelowania - Blenderownia v6.0[1913], Drewniane koło (podstawy modelowania) - Blenderownia v6.0[1914], Różne materiały na jednym obiekcie - Blenderownia v6.0[1915], Neonówka - Blenderownia v6.0[1916], Blender 3D: Blending Into Python[1917], Blender 3D: Noob to Pro[1918].

Edycja audio-wideo

Cinelerra

Cinelerra jest programem do montażu wideo, umożliwia składanie filmu, dodawanie efektów itp. W przypadku problemów z uruchamianiem (biblioteki) polecam sprawdzenie wersji bibliotek faad i faac (najlepiej stosować z tego samego źródła co sama cinelerra).

1) zaznaczamy interesujący nas fragment (w oknie głównym)

2) w menu prawego przycisku myszy wybieramy Attach effect

3) jego ustawienia znajdziemy w tym menu pod pozycją Show

Efekty przejść

Wstawiamy je przeciągając wybrany efekt z okna efektów do okna głównego i umieszczając na połączeniu dwóch fragmentów. W menu prawego przycisku myszy mamy czas trwania i ustawienia.

Eksport

Należy przewinąć film do początku (eksport zaczyna się od aktualnego czasu) następnie File -> Render … . Polecam ustawić "File Format" na RawDV oraz "Insertion strategy" na "Insert nothing" (nie testowałem ustawień z eksportem audio).

PAKIET TRANSCODE I INNE NARZĘDZIA TEKSTOWE

AUDIO

wydzielenie ścieżki audio z pliku wideo:
`tcextract -a 0 -i "IN.avi" -x mp3 > OUT.mp3`
docinanie plików audio (program wywala sie przy generowaniu ostatniego pliku ale to niewielka przeszkoda):
`tcmp3cut -i 1.mp3 -o 1a.mp3 -t 330000,1725999 -d 0`
nagrywanie audio:
`rec -t ogg -d /dev/dsp OUT.ogg` (oczywiscie najpier ustawic odpowiednio pierwsze wejscie w mixerze ...)
łączenie plików audio (mp3):
`mp3wrap all.mp31.mp32.mp33.mp34.mp3`
poprawianie pliku mp3: (metoda brutalna ale skuteczna ...)
`lame --decode all_MP3WRAP.mp3 ; lame -b64 -h all_MP3WRAP.mp3.wav`

WIDEO

docinanie plików wideo:
`avisplit -i IN.avi -o OUT.avi -t 00:05:30-43175`
odzyskanie informacji o długosci filmu:
`aviindex -i broken.avi -n -o broken.idx; avimerge -x broken.idx -i broken.avi -o fixed.avi`

w przypadku problemow z audio warto z orginalu wydzielic sciezke audio a nastepnie polaczyc poprawione video i wydzielone uprzednio audio
kodowanie xvid:

pierwszy przebieg:
`transcode -i IN.avi -o TMP.avi -y xvid4 -R 1`

pierwdrugi przebieg:
`transcode -i TMP.avi -o OUT.avi -y xvid4 -R 2 -w 3500,25,100 -avi_limit 4000`
konwersja plików .ts (zapisu transmisji DVB) na .mp4 (potrafi naprawić identyfikację kanału oraz przwijanie):
`avconv -i IN.ts -codec copy OUT.mp4`

AUDIO I WIDEO

łączenie plików wideo i pliku audio ze sobą:
`avimerge -i 1.avi2.avi3.avi4.avi -p all_MP3WRAP.mp3.wav.mp3 -o calosc.avi -A 0`
poprawianie pliku wideo dla dźwięku:
`transcode -i calosc_video.avi -P 1 -x avi,null -y raw,null -o calosc.avi -c 0-140831` (uwaga: transcode ma problem z autodetekcją typu pliku gdy podany parametr -p)
przekodowanie tylko ścieżki audio: (bez ruszania wizji)
`transcode -i IN.avi -P 1 -b 56 -s 4 -o OUT.avi -y raw`

LINKI

Zobacz w Sieci: scalanie napisów z filmem[1919], Signal Analysis/Processing Software[1920].

Uwagi konfiguracyjne dotyczące multimediów

Tuner radiowo-telewizyjny i cyfrowa rejestracja wideo

Uruchomienie karty telewizyjnej wymaga zapoznania się z listą obsługiwanych kart i tunerów, odnalezienia na niej posiadanej pozycji bądź pozycji zbliżonej a następnie załadowaniu odpowiedniego modułu z odpowiednimi parametrami. W moim wypadku (karta PV-BT878P+ rev.2F z tunerem TPI8PSB02P) słuszne okazało się `modprobe bttv card=138 tuner=28 pll=1 radio=1`. Następnie możemy przystąpić do konfiguracji programu do odbioru telewizji / radia (np. xawtv, fbtv); przykładowy plik z konfiguracją dla AsterCity (nazwy niektórych kanałów mogą być nieaktualne) - .xawtv[1921] (jeżeli istnieje /etc/xawtv radzę zarchiwizować i skasować). Przy karcie graficznej nvidii spotkałem się z problemem polegającym na niewyświetlaniu obrazu przez xawtv - rozwiązaniem okazało się `xawtv -noxv` (aby ustalić ten parametr jako domyślny należy do pliku /.Xdefaults dodać "xawtv.xv: no").

Do nagrywania cyfrowego wideo polecam program "transcode" (niestety nie ma go w oficjalnym Debianie) z następującymi parametrami `transcode -i /dev/video0 -p /dev/dsp0 -g 768x576 -b 64 -e 16000,16,1 -o "$1" -x v4l -y xvid4 -R 1 -avi_limit 2048` (rejestruje to obraz w najlepszej możliwej w PALu jakości). Wcześniej przy pomocy ulubionego programu nastawiamy żądaną stację, przy poprawnej pracy dysku i nie oglądaniu zwartości katalogu w którym są zapisywane pliki nie powinno gubić klatek.

Niekiedy na nagraniach w maksymalnej rozdzielczości PAL pojawia się problem przekłamanych linii (wynika to z emisji w PALu jednej klatki obrazu w postaci dwóch półklatek wyświetlanych jako jednej) warto wtedy w VLC z menu Stettings wybrać Prefernces - General settings - Obraz i ustawić Video filter module na "Deinterlacing video filter" (następnie zrestartować vlc).

Wspomniany już VLC umożliwia także przechwytywanie i wysyłanie w sieć IP sygnału audio-wideo z naszej karty TV (oraz z pewnym wspomaganiem (program radio służący do nastrojenia odbiornika) także sygnału audio z tunera radiowego). Warto pamiętać że kanał możemy zmieniać zewnętrznie (już po odpaleniu vlc) przy pomocy v4lctrl z pakietu xawtv. Wersja 1.0 VLC ma pewne problemy z obsługą niektórych urządzeń V4L, objawia się to błędem "v4l demux error: chroma selection failed" (dla v4l) lub "v4l2 demux error: cannot get video input characteristics" (dla v4l2). Rozwiązaniem (przynajmniej dla kart na bttv) okazuje się być dodanie parametru `chroma_agc=1` do modułu jądra.

VLC może również posłużyć do odbioru sygnału telewizji cyfrowej (DVB). Konieczne do tego jest oczywiście załadowanie stosownych modułów z odpowiednimi opcjami (dla testowanej karty "Medion" było to: `modprobe saa7134 i2c_scan=1 card=12 tuner=30; sleep 2; modprobe saa7134_alsa`). Potem wystarczy `vlc dvb:// --dvb-frequency=690000000 --dvb-budget-mode`, gdzie 690000000 jest częstotliwością multipleksu który chcemy odbierać, `dvb-budget-mode` powoduje odbieranie przez VLC całego strumienia (całego multiplexu, zawierającego zwykle kilka kanałów) i tym samym umożliwa jego rejestrację. Do DVB przydany może być także programik `dvbtune` (w szczególności `dvbtune -i -f 690000000` wyświetli informacje o kanałach na podanym multiplexie, a `dvbtune -m -f 690000000` umożliwia monitorowanie sygnału)

Niekiedy można spotkać się z problemem polegającym na braku sprzętowego sprzężenia wejścia liniowego z wyjściem głośnikowym - problem ten może rozwiązać program kopiujący wejście na wyjście karty dźwiękowej (wersja alsa), (wersja oss) oraz skrypt dla KDE uruchamiający cały zestaw aplikacji do słuchania radia / oglądania telewizji. Podobnym zagadnieniem jest obsługa kart transmitujących audio tylko po PCI - należy wtedy przekierować dźwięk z urządzenia dźwiękowego związanego z kartą RTV na główną kartę dźwiękową - `arecord -D hw:2,0 -r 32000 -c 2 -f S16_LE | aplay - & xawtv -c /dev/video1 -vbidev /dev/vbi1 && killall arecord`

Jako że składnia opisywanych programów do rejestracji i transmisji sieciowej sygnału multimedialnego nie koniecznie jest łatwa i przyjemna to zachęca do zapoznania się z moim projektem skrypu ułatwiającego te operacje. Na koniec warto wspomnieć także o programie alevt[1922], który umożliwia softwerowe dekodowanie teletextu (telegazety) z karty obsługiwanej przez bttv: `alevt -vbi /dev/vbi0 -charset latin-2`.

Serwer dźwięku

Z serwera dźwięku KDE potrafią korzystać też liczne programy nie stworzone specjalnie dla tego środowiska (jak VLC, XMMS, ...), aby było to możliwe konieczne jest zainstalowanie do nich wtyczki dla systemu dźwiękowego "arts" oraz wybranie tego systemu jako wyjścia dźwięku. Ma to tę zaletę że programy te nie blokują sobie wtedy wzajemnie oraz serwerowi dźwięku KDE dostępu do urządzenia dźwiękowego (co ma miejsce przy korzystaniu z

/dev/dsp). Jeżeli korzystamy z sterowników alsa możemy pokusić się o wykorzystanie miksera wbudowanego w ten system dźwięku (dmix) do miksowania dźwięków z serwera arts oraz innych programów (jest to działanie domyślne tego sterownika). ALSA umożliwia także bardziej dziwne sztuczki z karta dźwiękową jak np. zamiana kanałów (np. prawego z lewym lub głośników tylnych z przednimi) - tutaj[1923] zamieszczam przykład stosownego pliku konfiguracyjnego. Z plikiem tym poprawnie współpracuje VLC z dźwiękiem 5.1. Przy różnych eksperymentach z przekieroywaniem wyjść bardzo przydatny może okazać się `speaker-test -c6 -twav` umożliwiający testowanie poprawności ustawień (z opcjami `-tsine -f3000` ten sam program może posłużyć do genrowania dźwięku sinusoidalnego o zadanej częstości). Jeżeli mamy więcej niż 1 kartę dźwiękową warto zainteresować się parametrem modułów alsy `index=`, odpowiedzialnym za ustalanie kolejności kart w systemie.

Nie rzadko okazuje się iż możliwości konfiguracyjne alsy (asound.conf) czy też artsd są niewystarczające wtedy warto zainteresować się PulseAudio lub nawet jack. PulseAudio[1924] jest dość prostym serwerem dźwięku, jego dużą zaletą jest zaawansowany mixer (pavucontrol) umożliwiający regulacją parametrów niezależnie dla każdego podłączonego do serwera procesu. Przy jego uruchamianiu należy zwrócić uwagę na przynależność użytkownika do którego należą klienci do odpowiednich grup na których operuje serwer. Niestety nawet pomimo chyba pomyślnego odpalenia serwera z priorytetem RT (co samo w sobie też nie jest łatwe) wyraźnie zauważalne było rwanie się dźwięku.

Zdecydowanie lepszym rozwiązaniem jest stosowanie serwera JACK[1925]. Serwer ten pozwala na swobodne aranżowanie ścieżki którą przechodzi sygnał audio, dzięki czemu możliwe jest podłączenie mixera czy nawet procesora efektów pod wybrany program lub pod całość systemu. Serwer odpalony z priorytetem czasu rzeczywistego daje bardzo dobre efekty. Najprostszą metodą odpalenia jackd z szeregowaniem czasu rzeczywistego (jak dla mnie szybszą w wdrożeniu, pewniejszą, itd od zabaw z `/etc/security/limits.conf`), tak aby jednocześnie działał na koncie i grupie jakiegoś użytkownika (aby można było się do niego podłączać) jest użycie następującego wrapper'a (musi mieć ustawiony SETUID na root lub być odpalany z root'a):

```
#include <sched.h>
#include <sys/resource.h>
#include <sys/types.h>
#include <errno.h>
#define UID 1000
#define GID 1000
int main () {
        pid_t pid;
        struct sched_param sp;
        pid = getpid();
        sp.sched_priority = sched_get_priority_max(SCHED_FIFO) -30;
        if ( sched_setscheduler(pid, SCHED_RR, &sp) ) {
                perror("Blad uzyskania szeregowania RT");
                return -1;
        }
        setresgid(GID, GID, GID);
        setresuid(UID, UID, UID);
        execlp("jackd", "jackd", "-dalsa", "-dhw", 0);
}
```

Przy korzystaniu z serwera jack warto zainteresować się aplikacjami takimi jak: `qjackctl` (graficzny manager połączeń kanałów audio), `jamin` (regulator), `jackeq` (regulator), `jack-rack` (zbiór efektów itp). Możliwe jest także wymuszenie kierowania audio z programów nie potrafiących korzystać z jakcd do niego za pomocą `/etc/asound.conf`:

```
pcm.jack {
        type jack
        playback_ports {
                0 alsa_pcm:playback_1
                1 alsa_pcm:playback_2
        }
        capture_ports {
                0 alsa_pcm:capture_1
                1 alsa_pcm:capture_2
        }
}
```

Możliwe jest nawet podłączenie pulseaudio do jackd, wymaga to odpalenia pulseaudio z odpowiednim konfigiem w którym najistotniejsze jest przekierowanie wyjścia do jackd za pomocą załadowania odpowiednich modułów (zamiast modułów dostępu do rzeczywistego sprzętu):

```
load-module module-jack-sink
load-module module-jack-source
```

Jako że współczesne karty nie często obsługują sprzętowo MIDI[1926] to godnym zainteresowania wydaje się programowy serwer syntezujący MIDI - np. TiMidity++[1927].

Zobacz w Sieci: ALSA - Kontrola głośności per aplikacja - HOW TO[1928], Audio mixer[1929], Applications using JACK[1930], HOWTO: PulseAudio Fixes & System-Wide Equalizer Support[1931]

VLC niestandardowo

VLC[1932] jest dość popularnym odtwarzaczem plików multimedialnych. Jednak jego możliwości nie kończą sie na zwykłym odtworzeniu pliku .avi - umożliwia on obsługę strumieni sieciowych (zarówno odbiór jak i nadawanie), odtwarzanie wielu źródeł w ramach jednego okna (a także strumieniowanie takiego sygnału), obsługę urządzeń V4L (kart telewizyjnych), DVB, DVD, … . VLC posiada także, przydatną przy oglądaniu strumieni sieciowych lub telewizyjnych, funkcję nagrywania odbieranego sygnału. Domyślnie jeżeli wejściowy sygnał ma postać skompresowaną (np. DVB) to rejestrowany jest bez rekompresji (warto zauważyć, iż domyslnie rejestrowany jest tylko aktualnie oglądany program, natomiast gdy VLC odpalony jest z opcją `--dvb-budget-mode` rejestrowany jest cały multiplex). Poniżej prezentuje kilka przykładowych konfigów i skryptów wykorzystujących ponadstandardowe zdolności VLC.

- tv.vlc.sh - skrypt do oglądania telewizji i słuchania radia z wykożystaniem VLC (używa dcop, konsole z KDE oraz set_station.sh do ustawiania kanału w oparciu o konfig xawtv)
- rtv.sh - skrypt do nagrywania i strumieniowania sygnału RTV wykożystujący m.in. VLC

Wtyczka dla przeglądarek Mozilla

Możemy osadzić odtwarzacz VLC na stronie WWW dzięki Mozilla plugin[1933]. Do tego celu można użyć następującego kodu:

```
<object id="filmik1" data="URL://pliku.lub.strumienia/ktory.chemy.wyswietlic"
        type="application/x-vlc-plugin" width="426" height="316"
></object>
```

Takim obiektem możemy sterować przy pomocy javascript-u - np. umieścić podpisy oparte o wartość atrybutu id (skrypt szczególnie przydatny gdy na jednej stronie mamy kilka takich obiektów):

```
<script language="Javascript">
        function putsNames() {
                var vlc_video_entry = document.getElementsByTagName("object");
                for(i=0; i < vlc_video_entry.length; i++) {
                        vlc_video_entry[i].video.marquee.text =
↪  vlc_video_entry[i].id;
                        vlc_video_entry[i].video.marquee.position = "TOP";
                        vlc_video_entry[i].video.marquee.opacity = 255;
                        vlc_video_entry[i].video.marquee.color = 579329;
                        //vlc_video_entry[i].video.marquee.size = 2;
                        vlc_video_entry[i].video.marquee.enable();
                }
        }
</script>
```

VLM

VLM jest metodą konfigurowania VLC do bardziej zaawansowanego streamingu czy też odtwarzania. Umożliwia także realizację usług typu wideo na życzenie. Uruchamianie VLC z taką konfiguracją odbywa się z użyciem opcji `--vlm-conf`. Aby uniknąć dziwnych błędów niekiedy lepiej (bezpieczniej) jest robić to w taki sposób: `LANG=C cvlc --vlm-conf plik.conf`

VLM można użyć także do streamingu video z kilkoma ścieżkami dźwiękowymi poprzez następujący konfig:

```
new source1 broadcast enabled
setup source1 input "file:///sciezka/do/pliku.wideo"
setup source1 output #transcode{vcodec=mp2v,vb=10000,acodec=mpga}:bridge-in{id-offset=100}:std{access=udp{ttl=1},mux=ts,dst=226.1.226.1:1234}
new source2 broadcast enabled
setup source2 input "file:///sciezka/do/pliku.dodatkowego.audio"
setup source2 output #duplicate{dst=bridge-out{id=2},select=audio}
control source1 play
control source2 play
```

Konfig VLM można także obudować tak aby stanowił plik wykonywalny uruchamiający VLC z odpowiednią konfiguracją (poprzez dodanie jako pierwszej linii `#!/usr/bin/cvlc --vlm-conf` oraz nadanie prawa wykonywalności). Przykład takiej konfiguracji zamieszczam w plikach streaming.sh[1934] i mosaic.sh[1935].

W pewnych przypadkach warto zainteresować się strumieniowaniem z użyciem C++ RTMP Server[1936].

TV.VLC.SH

```
# Program demonstruje wykorzystanie komunikacji poprzez DCOP w prostym skrypcie
# shellowym uruchamiającym zestaw programów do słuchania radia / ogladania telewizji.
# tryb pracy rozpoznawany jest automatycznie po nazwie wywoływanego programu
# (np. uzytego dwoiązania symbolicznego)

# DCOP jest protokołem komunikacji międzyprocesowej wykorzystywanym w KDE
# umożliwia on sterowanie działającymi aplikacjami KDE (np. konsolą czy
# konquerorem) z innej aplikacji, a w szczególności poprzez program dcop z
# skryptów shellowych. Umożliwia nam to budowanie w oparciu o KDE własnych
# "aplikacji".

# Więcej o DCOP (a także wykorzystaniu dialogów KDE w skryptach bashowych -
# kdialog) w pomocy KDE -    help:/khelpcenter/userguide/kde-diy.html#dcop

# analiza nazwy z którą zaostał odpalony skrypt => wybór trybu radio/tv/...
MODE=`basename "$0"`
MODE=${MODE%.vlc.sh}
case $MODE in
 "tv"|"radio")
  VIDEO_DEV="/dev/video0"
  AUDIO_DEV="alsa://hw:0,0"

  # poczatkowe ustawienia glosnosci
  amixer set 'Input Source',0 Line
  amixer set 'Capture',0 30% on
  amixer set 'Digital',0 30% on
  # w powyzszym nie dziala on ale to `amixer cset numid=10 on` lub ponizsze (rownowazne) dziala:
  amixer cset iface=MIXER,name='Capture Switch',index=0 on

  # ustawiamy sprzatanie po skrypcie - wyciszenie
  trap "amixer set \"Capture\",0 0 off; amixer cset iface=MIXER,name=\"Capture Switch\",index=0 off;" 0

  if [ "$MODE" = "radio" ]; then
   VLC_OPT="--qt-start-minimized --no-video"
  else
   VLC_OPT="--video-on-top --deinterlace 1"
  fi
  ;;
 "tv2")
  VIDEO_DEV="/dev/video1"
```

```
  AUDIO_DEV="alsa://hw:2,0"

  SET_STATION_OPT="-d /dev/video1"
  VLC_OPT="--video-on-top --deinterlace 1"
  MODE="tv"
  ;;
 *)
  echo "Unknown MODE"
  exit
  ;;
esac

function start_vlc() {
 vlc $VLC_OPT --no-fullscreen v4l2://$VIDEO_DEV :v4l2-norm=pal --v4l2-width=640 --v4l2-⤸
height=480 :v4l2-input=0 \
  :input-slave=$AUDIO_DEV :audio=1 >/dev/null 2>/dev/null
}

if [ -e "/usr/share/services/konsole-script.desktop" -o -e "~/.kde/share/services/konsole-⤸
script.desktop" ]; then
 konsole=$(dcopstart konsole-script)

# pobieramy identyfikator i modyfikujemy nazwe bierzace sesji
 session=$(dcop $konsole konsole currentSession)
 dcop $konsole $session renameSession Głosność
# wylaczamy zapamietywanie histori basha
 dcop $konsole $session sendSession 'HISTFILE=""; export HISTFILE;'
dcop $konsole $session sendSession 'alsamixer -V all'
dcop $konsole $session sendSession ''

# uruchamiamy nowa sesje, pobierajac jej identyfikator, zmieniamy jej nazwe i odpalamy ⤸
 set_station.sh
 session=$(dcop $konsole konsole newSession)
 dcop $konsole $session renameSession SetStation
dcop $konsole $session sendSession 'HISTFILE=""; export HISTFILE;'
dcop $konsole $session sendSession "set_station.sh -m $MODE $SET_STATION_OPT $@"
dcop $konsole $session sendSession ''

# przenosimy okienko na 3 pulpit wirtualny
 wmctrl -i -r `dcop $konsole konsole-mainwindow#1 getWinID` -t 3
 # i przelaczamy się na niego
 wmctrl -s 3

# start VLC
 start_vlc

# zamknięcie konsoli po zakończeniu VLC
 dcop $konsole MainApplication-Interface quit
elif [ -x "`which xfce4-terminal`" ]; then
 xfce4-terminal --disable-server --title "volume"  -e "alsamixer -V all" \
                         --tab  --title "station" -e "set_station.sh -m $MODE ⤸
                         $SET_STATION_OPT $@" &
 start_vlc
kill `jobs -p`
else
 echo "Couldn't find supported terminal emulator"
start_vlc
fi
```

SET_STATION.SH

```
#!/bin/bash


#
# DEFAULT SETTINGS AND HELP
#

MODE="tv"
STATION_LIST=".set_station.cfg"
DEF_DEV_TV="/dev/video0"
DEF_DEV_RADIO="/dev/radio0"

print_help() {
 echo "USAGE:"
echo "   $0 [-s STATION] [-d DEVICE_PATH] [-c CONFIG_FILE] [-m tv|radio]"
echo "   $0 [--station STATION] [--device DEVICE_PATH] [--config CONFIG_FILE] [--mode ↩
 tv|radio]"
echo "   $0 -h"
echo "   $0 --help"
echo " default is tv mode"
echo " deafult DEVICE_PATH is $DEF_DEV_TV for TV and $DEF_DEV_RADIO for radio"
echo ""
echo " if is set XAWTV_COMPATIBILITY enviroment variable script reading STATION names from:"
echo "   ~/.xawtv and ~/.radio config file"
echo " otherwise read config file from option or default ~/$STATION_LIST"
echo ""
echo "CONFIG FILE FORMAT IS:"
echo " MODE FRENQUENCY_IN_MHz STATION_NAME"
echo "   MODE = radio | tv"
echo ""
echo "Conversion to .xawtv format:"
echo " awk 'BEGIN {printf(\"[defaults]\nnorm = PAL\n\n\")} /^tv/ {printf(\"[%s]\nfreq = ↩
 %s\n\n\", $3, $2)}'"
exit
}

STATION_LIST="$HOME/$STATION_LIST"


#
# PARSING SCRIPT OPTIONS
#

eval set -- "`getopt -o hs:d:c:m: -l help,station:,device:config:mode: -- "$@"`"
while true; do
 case $1 in
  -h|--help)
   print_help
   ;;
  -s|--station)
   STATION=$2
   CURRENT=" --default-item \"$STATION\""
   ADDIONAL="1 \"$STATION\""
   shift
   ;;
  -d|--device)
   DEVICE=$2
   shift
```

```
    ;;
   -c|--config)
    STATION_LIST=$2
    shift
    ;;
   -m|--mode)
    MODE=$2
    shift
    ;;
   --)
    shift; break
    ;;
   *)
    shift
    ;;
  esac
done

MODE=${MODE:-"tv"}
if [ "$MODE" == "radio" ]; then
 DEVICE="${DEVICE:-$DEF_DEV_RADIO}"
else
 DEVICE="${DEVICE:-$DEF_DEV_TV}"
fi


#
# GETTING CHANNEL LIST    AND    DEFINE CONTROLING V4L DEVICE FUNCTION
#

#
if [ "${XAWTV_COMPATIBILITY-"no"}" == "no" ]; then
 CHANNEL_LIST=$(
  awk -v MODE=$MODE '$1==MODE {printf("\"%s\" \"%s\" ", $3, $2)}' < $STATION_LIST
 )
 function set_station() {
  freq="$( awk -v MODE=$MODE -v STATION=$1 '$1==MODE && $3 == STATION {print $2}' ⤸
  $STATION_LIST )"
  input=`expr $freq : 'input=\(.*\)'`
  if [ "$input" != "" ]; then
   v4l2-ctl -d $DEVICE --all --set-input=$input > /dev/null
  else
   v4l2-ctl -d $DEVICE --all --set-input=0 --set-freq=$freq > /dev/null
  fi
 }
else
 if [ "$MODE" == "radio" ]; then
  TMP_FILE_RADIO=`mktemp /tmp/set_station_radio.XXXXX`
  trap "rm $TMP_FILE_RADIO" 0
  CHANNEL_LIST=$(
   grep -v '^#' ~/.radio | awk '
    BEGIN {RS="["; FS="]"}
    $1=="Stations" {print $0}
   ' > $TMP_FILE_RADIO
   awk '
    BEGIN {FS="="}
    $2=="" {next}
    {printf("\"%s\" \"%.2f\" ", $2, $1/1000000)}
   ' $TMP_FILE_RADIO | sort -n
  )
```

```
  function set_station() {
   radio -q -c $DEVICE -f $(
    awk '
     BEGIN {FS="="}
     $2=="'$1'" {printf("%s",  $1/1000000)}
    ' $TMP_FILE_RADIO
   )
  }
 else
  CHANNEL_LIST=$(
   grep -v '^#' ~/.xawtv | awk '
    BEGIN {RS="["; FS="]"}
    $1=="" || $1=="global" || $1=="defaults" {next}
    {printf("\"%s\" \"%s\" ",  $1, $1)}
   '
  )
  function set_station() {
   v4lctl -c $DEVICE setstation "$1"
  }
 fi
fi


#
#PRESET STATION
#
[ "$STATION" != "" ] && set_station "$STATION"


#
# RUN DIALOG
#

TMP_FILE=`mktemp /tmp/set_station.XXXXX`
trap "rm $TMP_FILE" 0

while true; do
 eval dialog $CURRENT --menu CHANELL 0 0 0 $ADDIONAL $CHANNEL_LIST 2> $TMP_FILE || break;

PREV_STATION_2=$PREV_STATION_1
 PREV_STATION_1=$PREV_STATION_0
 PREV_STATION_0=$STATION

STATION=`cat $TMP_FILE`

if [ "$PREV_STATION_0" != "" ]; then
  if [ "$PREV_STATION_1" != "" ]; then
   ADDIONAL="1 \"$PREV_STATION_0\" 2 \"$PREV_STATION_1\""
  else
   ADDIONAL="1 \"$PREV_STATION_0\""
  fi
 fi

if [ "$STATION" = "1" ]; then
  STATION=$PREV_STATION_1
 elif [ "$STATION" = "2" ]; then
  STATION=$PREV_STATION_2
 fi

CURRENT=" --default-item \"$STATION\""
```

```
set_station "$STATION"
done
```

Transmisja i rejestracja RTV

Prezentuje skrypt służący do prostszego uruchamiania transmisji lub rejestracji sygnału RTV. Warto zapoznać się także z komendami wykorzystywanymi przez skrypt, gdyż jest całkiem prawdopodobne iż może być potrzebne ich wywołanie z innymi opcjami (np. rejestracja obrazu odbieranego przez sieć).

Alternatywna motoda rejestracji sygnału z karty telewizyjnej może byc taka funkcja którą możemy dopisac do `~/.bashrc`:

```
function tv_record() {
        if [ $# -lt 1 ]; then
                echo "USAGE: $0 STATION"
                echo "USAGE: $0 STATION /dev/videoX hw.X"
                return -1
        fi
        STATION="$1"
        VIDEO="${2:-/dev/video1}"
        AUDIO="${3:-hw.2}"
        v4lctl -c $VIDEO setstation $STATION
        mencoder tv:// -tv device=$VIDEO:input=0:norm=PAL-DK:adevice=$AUDIO:alsa:input=0:amode=1:immediatemode=0:decimation=1 -ffourcc XVID -ovc lavc -lavcopts vcodec=mpeg4:vbitrate=3000:keyint=125:vhq -oac mp3lame -lameopts cbr:br=96:mode=0 -o tv_`date +%F_%T`.avi
}
```

Z użyciem rtmpsuck możliwe jest także nagrywanie strumieni RTMP (np. filmików z rutube), posłużyć do tego może komenda: `iptables -t nat -A OUTPUT -p tcp --dport 1935 -m owner \! --uid-owner root -j REDIRECT; rtmpsuck`

RTV.SH

```
#!/bin/bash

# skrypt jest dość wygodnym magnetowidem cyfrowym/stramerem sieciowym umożliwiającym proste uruchamianie
# programu transcode/vlc/mencoder z zadanymi parametrami,
# dodatkowo ustawia inne parametry jak kanał, mikser do nagrywania audio, ...


print_help_and_exit() {
 echo "USAGE: $0 [-a alias] [-e engine] [-i input_source] [-r|-t] [-f frenquency|-c chanell_name]"
echo "          [-d dev_num] [-q hq|vhs] [-l ttl] [-p] [-v] [-n] [output]"
echo " -e = set engine:"
echo "      vlc (default)"
echo "      transcode (tv/composite recording only)"
echo "      arecord (radio recording only)"
echo "      mplayer OR mencoder"
echo
 echo " -r = recorder"
echo " -s = network streaming (default)"
echo
 echo " -i = set input mode:"
echo "      radio (requie -f, frenquency in MHz),"
```

```
echo "            use this option in VLC for only /dev/dsp capture and ignore v4l errors"
echo "      tv (requie -f, frenquency in kHz)"
echo "      composite (default)"
echo " -d = set video device {NUM} for /dev/video{NUM} (default 0)"
echo " -D = set audio device {NUM} for /dev/dsp{NUM} (default empty)"
echo
 echo " -f = set frenquency for tv or radio station (in kHz for TV, in MHz for RADIO)"
echo " -c = set chanel name from ~/.xawtv for tv station"
echo
 echo " -q = set quality support in transcode recorder modes (default hq)"
echo " -t = set recording time in transcode recorder modes (default infinty)"
echo
 echo " -p = preview"
echo " -v = verbose"
echo " -n = no daemon"
echo
 echo " -h = help (this info)"
echo
 echo "output is "
echo "  IP.ADDRES:PORT for network streaming mode (for example: 226.1.226.1:1234)"
echo "  file name without extension for recorder mode"

exit
}

# default value
ENGINE="vlc"
INPUT="composite"
MODE="NET"
VDEV_NUM=0
ADEV_NUM=""
ACSS="udp{ttl=1}"
QUALITY="hq"

# $FREQ nie może być pusty
FREQ="123"

while getopts "e:rsi:d:D:f:c:qtpvnh" arg; do
 case "$arg" in
  a)
   case "$OPTARG" in
    "alias1")
     ENGINE="vlc"
     INPUT="composite"
     MODE="NET"
     DEV_NUM=0
     ACSS="udp{ttl=1}"
    ;;
   esac
   ;;
  e) ENGINE="$OPTARG" ;;
  r) MODE="REC" ;;
  s) MODE="NET" ;;
  i) INPUT="$OPTARG" ;;
  f) FREQ="$OPTARG" ;;
  c) CHANELL="$OPTARG" ;;
  d) VDEV_NUM="$OPTARG" ;;
  D) ADEV_NUM="$OPTARG" ;;
  q) QUALITY="$OPTARG" ;;
  t) TIME="$OPTARG" ;;
```

```
  l) ACSS="udp{ttl=$OPTARG}" ;;

  p) PREVIEW="dup" ;;
  v) DEBUG="-vvv --color" ;;
  n) DAEMON="no" ;;
  h) print_help_and_exit ;;
 esac
done

# destination IP addres IP:PORT, for example (multicast): 226.1.226.1:1234
shift $(( $OPTIND-1 ))
OUTPUT=$1

if [ "$OUTPUT" = "" ]; then
 if [ "$MODE" = "REC" ]; then
  OUTPUT="rtv.`date +%FT%T`"
else
  print_help_and_exit
fi
fi


# ustawianie parametrów AUDIO
set_audio() {
 # USTAWIENIA GLOSNOSCI
 amixer set 'Input Source',0 Line
amixer set 'Capture',0 30% on
amixer set 'Digital',0 30% on
# w powyzszym nie dziala on ale to ‘amixer cset numid=10 on‘ lub ponizsze (rownowazne) ⤸
 dziala:
 amixer cset iface=MIXER,name='Capture Switch',index=0 on

# unmute audio
 v4lctl volume mute off
}

# uzyskanie częstotliwości z ~/.xawtv
get_FREQ() {
 awk '
  BEGIN {
   RS="["; FS="]"
  }
  $1=="'$1'" {
   print $0
  }
 ' .xawtv | awk '
  BEGIN {
   FS="[\t ]*=[\t ]*"
  }
  $1=="freq" {
   print $2
  }
 ' | tr "," "."
}

# TRYBY NAGRYWANIA
# dla plikow o maksymalnej jakosci oferowanej przez PAL
REC_transcode_hq() {
 /usr/bin/transcode -x v4l2,v4l2 -i /dev/video$VDEV_NUM -p /dev/dsp$ADEV_NUM -g 768x576 \
  -n 0x1 -b 64 -e 16000,16,1 \
```

```
  -y xvid4,tcaud -R 1,$1.xvid.log -o $1.avi $2
  # moznaby dodac -w 700,250,100 ale przy tym kodeku to bez (większego) znaczenia
}

# dla plikow o obnizonej jakosci (np. z VHS ...)
REC_transcode_vhs() {
 /usr/bin/transcode -x v4l2,v4l2 -i /dev/video$VDEV_NUM -p /dev/dsp$ADEV_NUM -g 384x286 \
  -n 0x1 -b 64 -e 16000,16,1 \
  -y xvid4,tcaud -R 1,$1.xvid.log -o $1.avi $2
  # -avi_limit 2048
  # -b 48 -y divx -w 450,60,100 -R 1 –export_fps 15
}

# nagrywanie przy pomocy mencoder'a
REC_mencoder() {
 mencoder tv:// \
  -tv device=/dev/video$VDEV_NUM:input=$V4L_INPUT:norm=PAL-DK:freq=$FREQ:⤸
  adevice=/dev/dsp$ADEV_NUM:immediatemode=0:decimation=1 \
  -ffourcc XVID -ovc lavc -lavcopts vcodec=mpeg4:vbitrate=3000:keyint=125:vhq \
  -oac mp3lame -lameopts cbr:br=96:mode=0 -o $1.avi
}


# dodatkowe ustawienia
if [ "$INPUT" = "composite" ]; then
 V4L_INPUT=1
else
 V4L_INPUT=0
 if [ "$CHANELL" != "" ]; then
  FREQ=`get_FREQ $CHANELL`
fi
fi
FREQ_kHz=`echo "$FREQ * 1000" | bc`


# wybór silnika i trybu
case "$ENGINE $MODE" in
 "vlc NET"|"VLC NET")
  #INPUT_V4L="v4l:///dev/video$VDEV_NUM :v4l-norm=pal :v4l-frequency=$FREQ :v4l-size=640x480 :⤸
  v4l-channel=$V4L_INPUT :v4l-adev=/dev/dsp$ADEV_NUM :audio=0"
  INPUT_V4L="v4l2:///dev/video$VDEV_NUM :v4l2-norm=pal :v4l2-tuner-frequency=$FREQ_kHz :v4l2-⤸
  size=640x480 :v4l2-input=$V4L_INPUT :input-slave=alsa://hw:2,0 :audio=1"
  OUTPUT_NET="std{access=$ACSS,mux=ts,dst=$OUTPUT}"

  CODEC="ffmpeg{keyint=80,hurry-up,vt=800000}"
  TRANSCODE="transcode{vcodec=mp4v,acodec=mpga,vb=3000,ab=256,venc=$CODEC}"
  DUPLICATE="duplicate{dst=display,dst=$OUTPUT_NET}"

  # for send to perrcast
  # OUTPUT_NET="std{access=http,mux=ogg,dst=$DEST}"
  # TRANSCODE="transcode{acodec=vorb,ab=128}"
  # and start via -r local_IP:local_port and use http://local_IP:local_port in peercast ⤸
  config
  # peercast may be on other machine

  if [ "$PREVIEW" = "dup" ]; then
   OUTPUT_VLC="#$TRANSCODE:$DUPLICATE"
   INTERFACE=""
   DAEMON="no"
  else
```

```
  OUTPUT_VLC="#$TRANSCODE:$OUTPUT_NET"
  INTERFACE="-I hotkeys"
 fi

 set_audio

 # URUCHAMIANIE VLC
 if [ "$INPUT" = "tv" -o "$INPUT" = "composite" ]; then
  vlc $DEBUG $INPUT_V4L --sout "$OUTPUT_VLC" $INTERFACE &
  sleep 1
  if [ "$CHANELL" != "" ]; then
   v4lctl setstation $CHANELL
  fi
 elif [ "$INPUT" = "radio" ]; then
  vlc $DEBUG $INPUT_V4L --sout "$OUTPUT_VLC" --no-video $INTERFACE &
  sleep 1
  radio -f $FREQ -q
 fi
 ;;
"transcode REC")
 if [ "$CHANELL" != "" ]; then
  v4lctl setinput Television
  v4lctl setstation $CHANELL
 elif [ "$INPUT" = "composite" ]; then
  v4lctl setinput Composite1
 else
  echo "transcode requies -i composite OR -c options"
  exit
 fi

 set_audio

 if [ "$TIME" != "" ]; then
  TIME="-c $TIME"
 fi

 REC_transcode_$QUALITY $OUTPUT $TIME
 ;;
"arecord REC")
 if [ "$INPUT" != "radio" ]; then
  print_help_and_exit
 fi

 set_audio
 radio -f $FREQ -q
 arecord -fcd $OUTPUT.wav
 ;;
"mplayer REC"|"mencoder REC")
 REC_mencoder $OUTPUT
 ;;
*)
 if [ "$MODE" = "NET" ]; then
  echo "Not supported network stream (-t) with this engine (-e $ENGINE)"
 else
  echo "Not supported recording (-r) with this engine (-e $ENGINE)"
 fi
 ;;
esac

# INFORMACJE O PID / ZAKANCZANIE PRACY
```

```
pid=`jobs -p`
if [ "$pid" != "" ]; then
 echo "DAEMON PID is $pid"
if [ "$MODE" = "NET" ]; then
  echo "for watch or listen use: vlc udp://@$OUTPUT"
  echo " OR: vlc udp://@$OUTPUT --sout '#duplicate{dst=std{access=file,mux=ogg,dst=\"record.↩
  ogg\"}}' -I hotkeys"
  echo " to record in record.ogg file"
fi

if [ "$DAEMON" = "no" ]; then
  echo -e "\033[1;31mfor exit write \033[1;0m\033[1;41mquit\033[0m"
  cmd=""
  while [ "$cmd" != "quit" ]; do
   read cmd
  done;
  kill $pid
 else
  echo -e "\033[1;31mfor exit exec \033[1;0m\033[1;41mkill $pid\033[0m"
fi
fi

# MORE INFO:
#    * http://www.videolan.org/doc/streaming-howto/en/ch03.html
#    * vlc --longhelp --advanced | less
#    * man radio
```

v4l2loopback

v4l2loopback jest modułem umożliwiającym utworzenie oszukanego urządzenia `/dev/video`, które pokazywałoby materiał zapisywany przez jakiś program (np. jakiś plik z filmem lub przechwytywamny obraz pulpitu - `vlc screen://`). Jest to szczególnie użyteczne gdy chemy taki materiał wstawić np. do wideokonferencji prowadzonej z uzyciem linphone lub innego programu używającego v4l jako źródła sygnału. Załączony do dokumentacji modułu przykład pokazuje jak można uzyć tego modułu, wymaga on jednak podania materiału w odpowiednim formacie surowego wideo. Przy eksperymentach z takimi formatami wideo pomocne mogą być komendy:

```
cvlc aaaa.mp4 --vout="YUV" --yuv-file="/tmp/temp.yuv" --no-yuv-yuv4mpeg2
 ↪  --yuv-chroma=I420
mplayer -rawvideo w=1280:h=720:fps=1:i420 temp.yuv
vlc --rawvid-width=1280 --rawvid-height=720 --rawvid-fps=1 temp.yuv --rawvid-chroma=I420
```

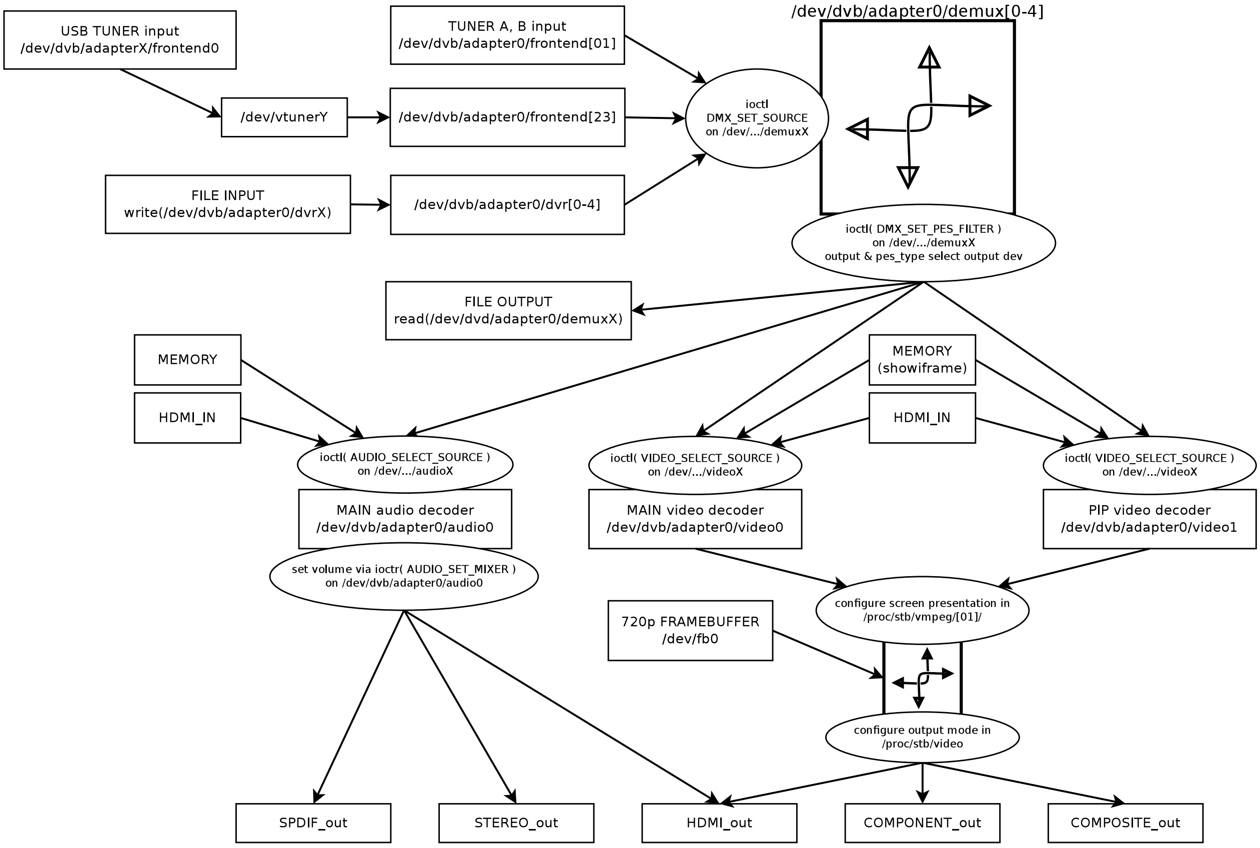
DVB STB: Enigma2

Enigma2 jest popularną aplikacją realizującą funkcjonalność dekodera telewizji cyfrowej w rozwiązaniach STB (set top box) opartych na Linuxie. Zajmuje się ona zarówno samą obsługą linuxowych urządzeń DVB - `/dev/dvb/adapert*/*` (czyli przełączaniem kanałów, źródeł sygnału, nagrywaniem sygnału kierowaniem go na wyjście wizyjne (np. HDMI) urządzenia), jak również obsługą GUI i wszelkich jego "feature'ów". Rdzeń aplikacji napisany jest w C++, natomiast elementy interfejsu użytkownika oraz różnego rodzaju rozszerzenia kodowane są w Pythonie.

Niestety w aplikacji nie został zastosowany podział na frontend (realizujący GUI) i backend (odpowiedzialny za obsługę DVB), efektem tego jest m.in. to że drobne błędy w GUI prowadzą do restartu całej aplikacji (i np. przerwania trwających nagrań). W porównaniu z innymi projektami FLOSS (Enigma2 korzysta z licencji GPL) także rozwój projektu jest dziwny - projekt jako taki nie ma zasadniczo strony domowej, głównego repozytorium kodu, ani niczego w tym stylu - rozwijany jest jako zestaw forków, utrzymywanych w ramach poszczególnych dystrybucji oprogramowania dla STB, które od czasu do czasu wymieniają pomiędzy sobą fragmenty kodu. Wygląda to tak jakby Linux nie miał swojego repozytorium kodu, struktur projektu, itd, zamiast tego każda dystrybucja miałaby własne repozytorium kodu jądra i od czasu do czasy wymieniała kod z innymi dystrybucjami. Nie udało mi się również

nigdzie odnaleźć dokumentacji pythonowego API enigmy. W przypadku dystrybucji enigmy zauważalne jest dość luźne (przynajmniej z punktu widzenia użytkownika Debiana) podejście do zagadnień licencyjnych - np. wiele z pluginów enigmy nie ma w ogóle określonej licencji (jest tylko kod w jakimś repozytorium), bardzo duża część ma licencje CC ”non-comercial” (a pomimo tego znajdują się w dystrybucjach które są preinstalowane w sprzedawanych urządzeniach).

Poniższe opisy bazują na moich doświadczeniach z Enigmą2 pochodzący z dystrybucji OpenATV[1937] oraz urządzeniu Golden Interstar Xpeed LX3 (ten sam sprzęt występuje także pod nazwami ”Sezam Marvel” i ”inihdp”). Ogólną strukturę urządzeń DVB w tym urządzeniu można przedstawić w sposób następujący:



Jako że OpenATV (dystrybucja Linuxa z Enigmą oparta, jak większość podobnych, na managerze pakietów ipkg/opkg) nie udostępnia pakietów deweloperskich, a cross-kompilacja (zwłaszcza na system dla którego nie mamy bibliotek w wersjach deweloperskich) bywa uciążliwa, a zasoby STB na to pozwalają to do kompilacji postanowiłem wykorzystać Debiana działającego w chroocie. Instalacja Debiana wykonana została poprzez debootstrap - pierwszy etap (z opcją `--foreign`) na PC, przekopiowanie tar-em na HDD STB i tam w chroocie drugi etap debootstrapem (opcja `--second-stage`).

Mini-rootfs

Celem wygodnego uruchamiania usług z (potencjalnie różnych) środowisk chrootowych oraz zamienienia samego openAVT na środowisko chrootowe przygotowany został na bazie OpenAVT minimalny rootfs. Składają się na niego:

— agetty
— busybox (podlinkowany jako: blkid cat chmod chroot cut echo expr find getty grep hostname ifconfig inetd ip mdev mkdir mount readlink rm rmdir sed sh sleep telnetd udhcpc umount which)
— busybox.suid (podlinkowany jako: login)
— halt.sysvinit (podlinkowany jako: halt reboot)
— init.sh - opcjonalna alternatywa dla init.sysvinit, gdybyśmy chcieli się execować do init w chroocie
— init.sysvinit (podlinkowany jako: init)
— killall5
— kmod (podlinkowany jako: modinfo modprobe rmmod)
— sdparm

Wszystkie te programy wraz z dolinkowaniami umieszczone zostały w `/bin`, natomiast `/sbin` jest dowiąza-

niem symbolicznym do `/bin`. W `/lib` (ld.so.1, libc.so.6, libcrypt.so.1, libnss_files-2.19.so, libnss_files.so.2, libz.so.1) oraz moduły jądra. Ponaddto w skład mini-rootfs wchodzą pliki konfiguracyjnymi:

—
— - główny skrypt startowy wywoływany przez init
— - uruchamianie usług w chroocie
— - skrypt wyłączania / restartowania systemu
— - funkcie pomocnicze oraz zmienne konfiguracyjne dla skryptów w init.d
— `/etc/init.d/loadHotPlugModules.sh` - pochodzący z OpenAVT
— `/etc/inetd.conf` z jednym wpisem: `23 stream tcp nowait root /bin/busybox telnetd`
— `/etc/mdev.conf` - pochodzący z OpenAVT
— `/etc/mdev/mdev-mount.sh` - pochodzący z OpenAVT
— `/etc/udhcpc.script` - pochodzący z OpenAVT, bazujący na orginalnym skrypcie z busybox skrypt z dodanym sprawdzaniem istnienia `/etc/resolv.conf` przed wpisywaniem do niego adresów
— `/etc/passwd` - pochodzący z OpenAVT
— `/etc/group` - pochodzący z OpenAVT

BOOT.SH

```
#!/bin/sh

. /etc/init.d/init.common


print "Mount kernel filesystems ..."
mount -t proc proc /proc
mount -t sysfs sysfs /sys
test -c /dev/ptmx || mknod -m 666 /dev/ptmx c 5 2
mkdir -p /dev/pts
mount -t devpts devpts /dev/pts -o gid=$TTY_GRP -o mode=$TTY_MODE


print "Set hostname to $HOSTNAME ..."
hostname $HOSTNAME


print "Prepare chroot ..."
prepareChroot $CHROOT
mkdir -p $CHROOT/run/media/root/logs/


print "Start mdev and mout HDD ..."
# for detail config about auto-mount see /etc/mdev/mdev-mount.sh
echo > /dev/mdev.seq
echo > /dev/.udev
echo "/sbin/mdev" >/proc/sys/kernel/hotplug
mdev -s


print "Start network and telnet ..."
ifconfig lo up
udhcpc -i eth0 -b -s /etc/udhcpc.script
ip address add $NET_IP dev eth0
ip route add default via $NET_GW dev eth0
inetd /etc/inetd.conf


print "Load video and frame buffer modules ..."
modprobe dvb
```

```
print "Set splash ..."
echo "$VIDEOMODE" > /proc/stb/video/videomode
chroot $CHROOT /usr/local/bin/showiframe /usr/local/share/bootlogo.mvi
sleep 1


print "Set fan control ..."
echo 2   > /proc/stb/fp/fan       # ON
echo 255 > /proc/stb/fp/fan_vlt   # FULL SPEED
echo 50  > /proc/stb/fp/fan_pwm   # FULL SPEED


print "Init framebuffer ..."
chroot $CHROOT  /usr/local/bin/FBinit
#chroot $CHROOT fbset -fb /dev/fb0 -xres 1280 -yres 720
#for i in `seq 1 9`; do
# chroot $CHROOT    /usr/local/bin/con2fb /dev/fb0 /dev/tty$i;
#done


print "Load kernel modules ..."
# Realtek RTL2832 + R820T USB DVB-T tuner
modprobe r820t; modprobe rtl2832; modprobe dvb_usb_rtl28xxu
# hotplug modules
/etc/init.d/loadHotPlugModules.sh
```

CHROOT.SH

```
#!/bin/sh

. /etc/init.d/init.common

startstop() {
 print "$2 $1 in $CHROOT chroot"
chroot $CHROOT /etc/init.d/$1 $2
}

case $1 in
 "start")
  startstop urandom $1
  startstop rsyslog $1
  startstop cron $1
  startstop ssh $1
  print "run ntpdate in $CHROOT chroot"
  chroot $CHROOT /usr/sbin/ntpdate-debian
  ;;
 "stop")
  sleep 1
  startstop ssh $1
  startstop cron $1
  startstop rsyslog $1
  startstop urandom $1
  ;;
 "getty")
  print "run getty in $CHROOT chroot"
  exec chroot $CHROOT /sbin/getty 38400 tty2
```

```
  ;;
 "FBconsole")
  print "run fbconsole in $CHROOT chroot"
  for i in 1 2 3 4 5; do
   [ -x /dev/vcsa2 ] || sleep 3
  done
  exec chroot $CHROOT /usr/local/bin/FBconsole
  ;;
 "enigma2")
  print "run enigma2 in $CHROOT chroot"
  exec chroot $CHROOT /usr/local/bin/enigma2
  ;;
 *)
  chroot $CHROOT $@
  ;;
esac
```

HALT.SH

```
#!/bin/sh

. /etc/init.d/init.common

sleep 1

print "Sending all processes the TERM signal..."
killall5 -15
sleep 5
print "Sending all processes the KILL signal..."
killall5 -9


print "Umount all filesystems (or remount to read-only)  ..."
umount -a -f -r
mount -o remount,ro / >&/dev/null


if [ $RUNLEVEL -eq 6 ]; then
 print "Reboot system ..."
reboot -d -f -i
else
 print "Stop SCSI devices ..."
sdparm --command=stop /dev/sd?
sleep 5
 print "Halt system ..."
halt -d -f -i -p -h
fi
```

INIT.COMMON

```
#
# init scripts configuration
#
TTY_GRP=5
TTY_MODE=620
```

```
HOSTNAME=xpeedlx3
CHROOT=/debian
VIDEOMODE=1080p
NET_IP="172.16.16.18/28"
NET_GW="172.16.16.17"

#
# init scripts common functions
#
print() { echo -e "\e[1;36m$1\e[0m"; }

prepareChroot() {
 mount -t proc  proc  $1/proc
mount -t sysfs sysfs $1/sys
mount -t tmpfs tmpfs $1/run
  mkdir $1/run/tmp $1/run/lock
  mkdir $1/run/log $1/run/cache $1/run/cache/debconf $1/run/media
  chmod 1777 $1/run/tmp $1/run/lock
  chmod 0755 $1/run/log $1/run/cache $1/run/cache/debconf $1/run/media
mount -o rbind /dev  $1/dev
}
```

INIT.SH

```
#!/bin/sh

REAL_INIT=/sbin/init.sysvinit


print() { echo -e "\e[1;36m$1\e[0m"; }

if [ -x /sbin/runlevel -a "`/sbin/runlevel`" != "unknown" ]; then
 print "Not unknown runlevel => Exec real init ..."
exec $REAL_INIT $@
fi

/etc/init.d/rcS.sh

print "End of init script => Exec real init ..."
exec $REAL_INIT $@
```

INITTAB

```
#
# default runlevel
#
id:3:initdefault:

#
# init and halt system
#
init::bootwait:/etc/init.d/boot.sh
halt:06:wait:/etc/init.d/halt.sh

#
# start and deamons in chroot
```

```
#
s10:234:wait:/etc/init.d/chroot.sh start
k10:016:wait:/etc/init.d/chroot.sh stop

#
# start "respawn" process
#
sc:234:respawn:/sbin/agetty --noclear --noreset console 115200
gc1:23:respawn:/sbin/agetty tty2
gc2:23:respawn:/etc/init.d/chroot.sh FBconsole
gui:34:respawn:/etc/init.d/chroot.sh enigma2
```

Building enigma2 z zależnościami

W "devel-chroot" instalujemy biblioteki developerskie

```
aptitude insall libcrypto++-dev libdvdnav-dev libfreetype6-dev libfribidi-dev libgif-dev
 ↪  libgstreamer-plugins-base0.10-dev libgstreamer0.10-dev libjpeg8-dev libpng12-dev
 ↪  libsdl1.2-dev libsigc++-1.2-dev  libxml2-dev libxslt1-dev
rm -fr /usr/include/linux/dvb
cp -a ~/deps/dvb_header_linux_3.14 /usr/include/linux/dvb
```

W "devel-chroot" kompilujemy i instalujemy enigma2 ecosystem (wykorzystywane patche można pobrać tutaj[1938]):

BUILD-CMD

```
# enigma2 dependencies - "libxmlccwrap", "libdvbsi++", "tuxtxt" and "boxbranding.so"
 mkdir -p /usr/local/share/enigma2 /usr/local/lib/enigma2 /usr/local/etc/enigma2
mkdir deps
(
  cd deps
  git clone git://git.opendreambox.org/git/obi/libxmlccwrap.git
  (
   cd libxmlccwrap
   autoreconf -i
   ./configure
   make
   make install
  )
  git clone git://git.opendreambox.org/git/obi/libdvbsi++.git
  (
   cd libdvbsi++
   autoreconf -i
   ./configure
   make
   make install
  )
  git clone git://git.code.sf.net/p/openpli/tuxtxt
  (
   cd tuxtxt/libtuxtxt
   touch AUTHORS ChangeLog NEWS README
   autoreconf -i
   ./configure --with-boxtype=generic --with-fontdir=/usr/local/share/fonts
   make
   make install
  )
  (
```

```
  cd tuxtxt/tuxtxt
  touch AUTHORS ChangeLog NEWS README
  autoreconf -i
  ./configure --with-boxtype=generic --with-configdir=/usr/local/etc/
  make
  make install
 )
 git clone http://github.com/oe-alliance/branding-module
 (
  autoreconf -i
  ./configure \
   --with-boxtype=xpeedlx3 \
   --with-machinebrand=GI \
   --with-machinename=LX-3 \
   --with-brandoem=ini \
   --with-machinebuild=inihdp \
   --with-distro=openatv \
   --with-mtdrootfs=mtd0 \
   --with-mtdkernel=mtd2 \
   --with-rootfile=rootfs.bin \
   --with-kernelfile=kernel.bin \
   --with-mkubifs="-m 4096 -e 1040384 -c 1984" \
   --with-ubinize="-m 4096 -p 1024KiB" \
   --with-driverdate="20140716" \
   --with-imageversion=4.1 \
   --with-imagebuild=0 \
   --with-imagedir=xpeedlx3 \
   --with-imagefs=ubi
  make
  make install
  rm /usr/local/lib/enigma2/python/boxbranding.la

  # compare results of branding-module/BoxBranding/lib/test.py running with:
  #     orginal boxbranding.so (/usr/local/lib/enigma2/python/boxbranding.so) and new build
  boxbranding.so
  # if need correct ./configure options
 )
# "enigma2"
 git clone http://github.com/openatv/enigma2 enigma2-openatv
(
 cd enigma2-openatv
 autoreconf -i
 ./configure --with-boxtype=xpeedlx3

 # build and run patches
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-build.diff
 # VT, keyboard, movie seek, epg, clock and debug info patches
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-VT.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-keyboard.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-seek.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-EPG.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-clock.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-debug.diff
 # test & dirty-hack patches
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-decoder-pesinput.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-RTL2832_R820T.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/Enigma2-dirty.diff
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/InfoBarGenerics-DirtyHacks.diff
 # plugins patches
 patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/InfoPanel.diff
```

```
  patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/DefaultServicesScanner.diff
  patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/DiseqcTester.diff

  echo "exit 0" > py-compile
  make
  make install
  ( cd data/vfd255 && make install )
  ( cd /usr/local/lib/enigma2/python/Plugins && rm -fr SystemPlugins/WirelessLan ⤸
  Extensions/DVDBurn Extensions/Modem Extensions/MediaScanner )

  wget -O /usr/local/lib/enigma2/python/process.py  https://raw.githubusercontent.⤸
  com/openembedded/openembedded/master/lib/oe/process.py
)
# skins
 mkdir skins
(
  cd skins
  git clone https://github.com/oe-alliance/oe-alliance-e2-skindefault skin-default
  (
   cd skin-default
   autoreconf -i
   ./configure
   make
   make install
  )
  git clone https://github.com/a4tech/mega32-egami.git mega32
  (
   cd mega32
   patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/SkinMega32.diff
   cp ~/NEW-FROM-SCRACH/rrp/patches/SkinMega32-channel.png ⤸
   usr/share/enigma2/mega32/skin_default/
   cp -a usr/share/enigma2/mega32 /usr/local/share/enigma2/
   cp -a usr/share/fonts/* /usr/local/share/fonts/
  )
 )
# tools
 mkdir tools
(
  cd tools

  git clone git://github.com/oe-alliance/aio-grab.git
  (
   cd aio-grab
   autoreconf -i
   ./configure
   make
   make install
  )

  mkdir showiframe && wget -O showiframe/showiframe.c https://raw.githubusercontent.⤸
  com/SIFTeam/openembedded/master/recipes/vuplus/vuplus-dvb-tools/showiframe.c
  (
   cd showiframe
   gcc showiframe.c -o showiframe
   cp showiframe /usr/local/bin/
  )

  cp -a arora-bin/qws /usr/local/
 )
# plugins
```

```
 mkdir plugins
mkdir plugins/e2openplugin
(
  cd plugins

  git clone http://github.com/E2OpenPlugins/e2openplugin-OpenWebif.git e2openplugin/openWebif
  (
   cd e2openplugin/openWebif
   patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/OpenWebif.diff
   cp -a plugin/ /usr/local/lib/enigma2/python/Plugins/Extensions/OpenWebif
  )

  git clone https://github.com/E2OpenPlugins/e2openplugin-StreamInterface.git ⤸
  e2openplugin/streamInterface    # unknow licence
#   cp -a e2openplugin/streamInterface/plugin/ ⤸
/usr/local/lib/enigma2/python/Plugins/Extensions/WebStreamInterface

  git clone https://github.com/E2OpenPlugins/e2openplugin-SimpleUmount ⤸
  e2openplugin/simpleUmount
  cp -a e2openplugin/simpleUmount/plugin/ ⤸
  /usr/local/lib/enigma2/python/Plugins/Extensions/SimpleUmount

  git clone https://github.com/E2OpenPlugins/e2openplugin-SetPicon e2openplugin/setPicon
#   cp -a e2openplugin/setPicon/plugin/ ⤸
/usr/local/lib/enigma2/python/Plugins/Extensions/SetPicon

  git clone https://github.com/E2OpenPlugins/e2openplugin-CrossEPG e2openplugin/crossEPG
#   (
#    cd crossEPG
#    ln -s /usr/include/libxml2/libxml /usr/include/
#    make
#    make install
#   )

  git clone https://github.com/betonme/e2openplugin-EnhancedMovieCenter.git ⤸
  enhancedMovieCenter
  (
   cd enhancedMovieCenter

   mkdir m4
   autoreconf -i
   ./configure
   echo "exit 0" > py-compile

   patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/EnhancedMovieCenter.diff
   make
   make install
  )

  cp -a ~/NEW-FROM-SCRACH/rrp/plugins/TelnetClient ⤸
  /usr/local/lib/enigma2/python/Plugins/Extensions/

  git clone http://github.com/oe-alliance/enigma2-plugins
  (
   cd enigma2-plugins
   autoreconf -i
   ./configure --with-boxtype=xpeedlx3
   echo "exit 0" > py-compile

   ( cd moviecut        && make ;  make install )
```

```
  ( cd vlcplayer       && make ;  make install )
  ( cd youtubeplayer   && make ;  make install )

  patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/bmediacenter.diff
  ( cd bmediacenter      && make ;  make install )  # unknow licence
  patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/oscaminfo.diff
  ( cd oscaminfo         && make ;  make install )  # unknow licence

  patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/aspectratioswitch.diff
  ( cd aspectratioswitch   && make ;  make install ) # non-free licence (no comercial), in ⤸
  plugin.py MIT-type
  ( cd epgsearch           && make ;  make install ) # non-free licence (no comercial)
  ( cd toolkit             && make ;  make install ) # non-free licence (no comercial), for ⤸
  epgsearch
 )

 git clone https://github.com/oe-alliance/oe-alliance-plugins
 (
  cd oe-alliance-plugins
  autoreconf -i
  ./configure --with-boxtype=xpeedlx3
  echo "exit 0" > py-compile

  ( cd AutoBouquetsMaker  && make && make install )
  ( cd Blindscan          && make && make install )
  ( cd TransCodingSetup   && make && make install ) # unknow licence
  patch -p1 < ~/NEW-FROM-SCRACH/rrp/patches/WebBrowser-arora.diff
  ( cd WebBrowser         && make && make install ) # unknow licence
 )
)
```

Instalacja enigma2 z zależnościami

W "production-chroot" przygotowujemy środowisko uruchomieniowe:

```
aptitude install libfreetype6 libxml2 libjpeg8 libfribidi0 libgif4 libgstreamer0.10-0
↪ ibgstreamer-plugins-base0.10-0 libpng12-0 libsigc++-1.2-5c2  libxslt1.1   llibgomp1
↪ libpython2.7
aptitude install python-twisted-core python-twisted-runner python-twisted-web python-usb
↪ python-openssl python-gdata
wget -O /tmp/python-cheetah_2.4.4-3+b1_mipsel.deb http://ftp.pl.debian.org/debian/pool/⌋
↪ main/c/cheetah/python-cheetah_2.4.4-3+b1_mipsel.deb && dpkg -i
↪ /tmp/python-cheetah_2.4.4-3+b1_mipsel.deb
wget -O /tmp/libssl0.9.8_0.9.8o-4squeeze14_mipsel.deb http://ftp.pl.debian.org/debian/⌋
↪ pool/main/o/openssl/libssl0.9.8_0.9.8o-4squeeze14_mipsel.deb && dpkg -i
↪ /tmp/libssl0.9.8_0.9.8o-4squeeze14_mipsel.deb
ln -s /usr/local/etc/enigma2      /etc/
ln -s /usr/local/etc/timezone.xml  /etc/
ln -s /usr/local/etc/tuxbox       /etc/
ln -s /usr/local/etc/tuxtxt       /etc/
ln -s /usr/local/lib/enigma2      /usr/lib/
ln -s /usr/local/share/enigma2    /usr/share/
ln -s /usr/local/share/fonts      /usr/share/
ln -s /usr/local/qws              /qws
```

Następnie należy skopiować `/usr/local` z deweloperskiego do produkcyjnego chroot'a. Aby nie zarzynać flasha warto także (wzorem orginalnego OpenATV) pewne katalogi przenieść do /run w którym montowany jest ramfs:

```
rm -fr /media /var/tmp /var/log /var/cache # /var/lock i /var/run są domyślnie linkami do
↪  /run
ln -s /run/media /media
ln -s /run/tmp   /var/tmp
ln -s /run/log   /var/log
ln -s /run/cache /var/cache
```

Narzędzia i rozszerzenia

TelnetClient

Jest to wtyczka do Enigmy implementująca funkcjonalność klienta telnet. Korzysta ona z pythonowego modułu terminal[1939]. Funkcjonuje zarówno w oryginalnej wersji enigmy2, jak i wersji zmodyfikowanej powyżej opisanymi łatkami. W nie zmodyfikowanej wersji enigmy modyfikatory klawiatury działają jednorazowy (czyli np. aby wpisać AB naciskamy shift, puszczamy go, wciskamy a, ponownie naciskamy i puszczamy shift, wciskamy b). Aby korzystać z wersji z usprawnieniami należy zmienić wartość `enigmaFixed` w `ConsoleKeyboardInput` na `True`. Całość wtyczki zaimplementowana jest w pliku: plugin.py

Pobierz[1940]

Narzędzia obsługi FrameBuffera

- FBinit.c[1941] - inicjalizacja framebuffera w Xpeed LX3
- FBclear.c - czyszczenie framebuffera
- FBconsole.c - przekierowanie konsoli z tty2 na framebuffer (zrobione trochę na około - tak aby nie korzystać z linuxowego rozwiązania konsoli na framebufferze bo nie chciało ono działać na testowanej wersji jądra/sprzęcie)

Narzędzia wspomagające edycję list kanałów

- userbouquet-names.sh - dodawanie do ”userbouquet” nazw kanałów w oparciu o lamedb w celu prostej edycji jako plik tekstowy

FBclear.c

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <inttypes.h>
#include <sys/mman.h>
#include <string.h>
#include <linux/fb.h>

#define DEV_FILE "/dev/fb0"

int    fbFD;
char*  fbBUF;
struct fb_var_screeninfo vinfo;
struct fb_fix_screeninfo finfo;
int    fbFrameSize, fbPixelSize;

inline void setARGB(int idx, uint8_t a, uint8_t r, uint8_t g, uint8_t b) {
 char* point = fbBUF + idx;
 *(point)   = b;
 *(point+1) = g;
 *(point+2) = r;
 *(point+3) = a; // 0 == transparent
}
```

```
inline void drawRectangle(int startX, int startY, int stopX, int stopY) {
 int x, y, lineOffet, pixelOffset;

if (stopX > vinfo.xres || stopX == 0)
  stopX = vinfo.xres;
 if (stopY > vinfo.yres || stopY == 0)
  stopX = vinfo.yres;
 if (startX < 0)
  startX = 0;
 if (startY < 0)
  startY = 0;

lineOffet = (vinfo.yoffset + startY) * finfo.line_length;
 for (y = startY; y < stopY; y++) {
  pixelOffset = lineOffet + (vinfo.xoffset + startX) * fbPixelSize;;
  for (x = startX; x < stopX; x++) {
   setARGB(pixelOffset, 150, 200, 200, 0);
   pixelOffset += fbPixelSize;
  }
  lineOffet += finfo.line_length;
 }
}

int main() {
 // open device and get info
 fbFD = open(DEV_FILE, O_RDWR);
 if (fbFD == -1) {
  perror("Open framebuffer device: " DEV_FILE " error");
  exit(1);
 }

if (ioctl(fbFD, FBIOGET_FSCREENINFO, &finfo) == -1) {
  perror("ioctl FBIOGET_FSCREENINFO error");
  exit(2);
 }

if (ioctl(fbFD, FBIOGET_VSCREENINFO, &vinfo) == -1) {
  perror("ioctl FBIOGET_VSCREENINFO error");
  exit(3);
 }

if (vinfo.bits_per_pixel != 32) {
  puts("We only support 32 bit per pixel");
  exit(4);
 }

// open memory buffer
 fbPixelSize = vinfo.bits_per_pixel / 8;
 fbFrameSize = vinfo.xres * vinfo.yres * fbPixelSize;
 fbBUF=(char*)mmap(0, fbFrameSize*2, PROT_WRITE, MAP_SHARED, fbFD, 0);

// write to frame buffer
 // drawRectangle(570, 250, 800, 550);
 memset(fbBUF, 0, fbFrameSize*2);

// close
 munmap(fbBUF, fbFrameSize);
 close(fbFD);
 return 0;
```

```
}
```

FBCONSOLE.C

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <inttypes.h>
#include <sys/mman.h>
#include <string.h>
#include <linux/fb.h>
#include <linux/vt.h>
#include <sys/ioctl.h>

#include <ft2build.h>
#include FT_FREETYPE_H
#include FT_GLYPH_H

#define DEV_FILE "/dev/fb0"

#ifndef FBIO_BLIT
#define FBIO_SET_MANUAL_BLIT _IOW('F', 0x21, __u8)
#define FBIO_BLIT 0x22
#endif

int    fbFD;
char*  fbBUF;
char*  fbBUFs[2];
struct fb_var_screeninfo vinfo;
struct fb_fix_screeninfo finfo;
int    fbFrameSize, fbPixelSize, fbBufNum;

void switchBuffers() {
 if (fbBufNum == 0) {
  // new render buffor is 1
  fbBufNum = 1;

  // display buffor 0
  vinfo.yoffset = 0;
 } else {
  fbBufNum = 0;
  vinfo.yoffset = vinfo.yres;
 }

// display old buffor
 ioctl(fbFD, FBIOPAN_DISPLAY, &vinfo);
 // set rendering to new buffor
 fbBUF = fbBUFs[fbBufNum];
}

inline void setARGB(int idx, uint8_t a, uint8_t r, uint8_t g, uint8_t b) {
 char* point = fbBUF + idx;
 *(point)   = b;
 *(point+1) = g;
 *(point+2) = r;
 *(point+3) = a;  // 0 == transparent
}
```

```
inline void drawRectangle(int startX, int startY, int stopX, int stopY) {
 int x, y, lineOffet, pixelOffset;

 if (stopX > vinfo.xres || stopX == 0)
  stopX = vinfo.xres;
 if (stopY > vinfo.yres || stopY == 0)
  stopX = vinfo.yres;
 if (startX < 0)
  startX = 0;
 if (startY < 0)
  startY = 0;

 lineOffet = (startY) * finfo.line_length;
 for (y = startY; y < stopY; y++) {
  pixelOffset = lineOffet + startX * fbPixelSize;;
  for (x = startX; x < stopX; x++) {
   setARGB(pixelOffset, 150, 150, 150, 150);
   pixelOffset += fbPixelSize;
  }
  lineOffet += finfo.line_length;
 }
}

int printCharBitmap(FT_GlyphSlot slot, int startX, int startY) {
 int fX, fY, fR, xOffset, lineOffet, pixelOffset;

 xOffset   = startX + slot->bitmap_left;
 lineOffet = (startY - slot->bitmap_top) * finfo.line_length;

 for(fY = 0; fY < slot->bitmap.rows; fY++) {
  pixelOffset = lineOffet + xOffset * fbPixelSize;
  fR = fY * slot->bitmap.width;

  for (fX = 0; fX < slot->bitmap.width; fX++) {
   if (pixelOffset > fbFrameSize)
    return 0;
   //setARGB(pixelOffset, slot->bitmap.buffer[fR + fX], 255, 0, 0);
   uint8_t val  = slot->bitmap.buffer[fR + fX];
   uint8_t rval = 255 - val;

   uint8_t a = (rval < 150) ? (255) : (150 + val);
   uint8_t r = (/* zero */ rval*150) >> 8;
   uint8_t g = (val*255  + rval*150) >> 8;

   setARGB(pixelOffset, a, r, g, r);

   pixelOffset += fbPixelSize;
  }
  lineOffet += finfo.line_length;
 }

 startX += slot->advance.x >> 6;
 return startX;
}

void renderText(char* text, FT_Face ftFace, int startY) {
 int          i, maxChar, xPos = 20;

 maxChar = strlen( text );
 for (i=0; i<maxChar; i++) {
```

```
    if ( FT_Load_Char(ftFace, text[i], FT_LOAD_RENDER) ) {
     puts("Error in FT_Load_Char");
     continue;
    }
    xPos = printCharBitmap(ftFace->glyph, xPos, startY);
    if (xPos == 0) break;
  }
}

#define DEVICE_SURFIX 2

int getConsole(char devSurfix, uint8_t *w, uint8_t *h) {
  int fd;
  char tmpBuf[20];

  snprintf(tmpBuf, 20, "/dev/vcsa%d", devSurfix);
  printf("Console on %s ", tmpBuf);

  fd = open(tmpBuf, O_RDONLY);
  if (fd < 0) {
    perror("open: ");
    exit(1);
  }
  if (read(fd, tmpBuf, 4) != 4) {
    perror("read: ");
    exit(2);
  }
  close(fd);

  *h = tmpBuf[0];
  *w = tmpBuf[1];
  printf(" has %d rows and %d cols\n", *h, *w);


  snprintf(tmpBuf, 20, "/dev/vcs%d", devSurfix);
  printf("Openning console on %s \n", tmpBuf);

  fd = open(tmpBuf, O_RDONLY);
  if (fd < 0) {
    perror("open: ");
    exit(3);
  }

  return fd;
}

int main(int argc, char** argv) {
  FT_Library    ftLib;
  FT_Face       ftFace;

  // FreeType open font
  if ( FT_Init_FreeType(&ftLib) )
    puts("Error in FT_Init_FreeType");

  if ( FT_New_Face(ftLib, "/usr/share/fonts/truetype/ttf-dejavu/DejaVuSansMono.ttf", 0, ⤸
  &ftFace) )
    puts("Error in FT_New_Face");

  if ( FT_Set_Char_Size( ftFace, 0, 10 * 64, 128, 128 ) )
    puts("Error in FT_Set_Char_Size");
```

```
// open device and get info
fbFD = open(DEV_FILE, O_RDWR);
if (fbFD == -1) {
  perror("Open framebuffer device: " DEV_FILE " error");
  exit(1);
}

if (ioctl(fbFD, FBIOGET_FSCREENINFO, &finfo) == -1) {
  perror("ioctl FBIOGET_FSCREENINFO error");
  exit(2);
}

if (ioctl(fbFD, FBIOGET_VSCREENINFO, &vinfo) == -1) {
  perror("ioctl FBIOGET_VSCREENINFO error");
  exit(3);
}

if (vinfo.bits_per_pixel != 32) {
  puts("We only support 32 bit per pixel");
  exit(4);
}

// open memory buffer (double buffering)
fbPixelSize = vinfo.bits_per_pixel / 8;
fbFrameSize = vinfo.xres * vinfo.yres * fbPixelSize;
fbBUFs[0] = fbBUF = (char*)mmap(0, fbFrameSize*2, PROT_WRITE, MAP_SHARED, fbFD, 0);
fbBUFs[1] = fbBUF + fbFrameSize;
fbBufNum = 0;
unsigned char reInit = 1;


int vcsFD, myFD;
struct vt_stat vtstat;

uint8_t width, height, y;
ssize_t sizeIN, sizeOUT;
char    *bufIN_1, *bufIN_2, *bufOUT, *bufTMP, charTMP;

vcsFD = getConsole(DEVICE_SURFIX, &width, &height);
myFD = open("/dev/tty0", O_RDONLY);

sizeIN  = width * height;
bufIN_1 = malloc(sizeIN);
bufIN_2 = malloc(sizeIN);
sizeOUT = (width+1) * height;
bufOUT  = malloc(sizeOUT);

while (1) {
  ioctl(myFD, VT_GETSTATE, &vtstat);
  if (vtstat.v_active == DEVICE_SURFIX) { // run only when DEVICE_SURFIX is active VT
    if (reInit == 1) {
      reInit = 0;
      memset(fbBUFs[0], 0, fbFrameSize*2);
      *bufIN_2 = '\0';
      ioctl(fbFD, _IOW('F', 0x21, __u8), &reInit); // disable FBIO_SET_MANUAL_BLIT
    }
    lseek(vcsFD, 0, SEEK_SET);
    if (read(vcsFD, bufIN_1, sizeIN) == sizeIN) {
```

```
    if ( strncmp(bufIN_1, bufIN_2, sizeIN) != 0 ) { // redraw only when changed VT output
     int renderY = 36;
     drawRectangle(16, 16, 906, 528);

     bufTMP = bufIN_1;
     for (y=0; y<height; y++) {
      // save first char of next line and put in its place "end of string"
      charTMP = bufTMP[width];
      bufTMP[width] = '\0';

      // show line
      renderText(bufTMP, ftFace, renderY);
      renderY += 20;

      // set bufTMP to start of next line and restore first char of next line
      bufTMP = bufTMP + width;
      *bufTMP = charTMP;
     }

     // swap bufIN_2 and bufIN_1
     bufTMP  = bufIN_2;
     bufIN_2 = bufIN_1;
     bufIN_1 = bufTMP;

     switchBuffers();
    }
   }
  } else {
   reInit = 1;
  }
  usleep(100000);
 }

FT_Done_Face(ftFace);
 FT_Done_FreeType(ftLib);

// close
 munmap(fbBUF, fbFrameSize);
 close(fbFD);
 return 0;
}
```

PLUGIN.PY

```
from enigma import eTimer, eActionMap, getPrevAsciiCode
from Screens.Screen import Screen
from Plugins.Plugin import PluginDescriptor
from Components.ActionMap import ActionMap
from Components.ScrollLabel import ScrollLabel
from keyids import KEYIDS
from Screens.MessageBox import MessageBox

from Components.Sources.StaticText import StaticText
from Components.Input import Input
from Screens.InputBox import InputBox

from Components.ConfigList import ConfigListScreen
```

```
from Components.config import ConfigText, ConfigInteger, getConfigListEntry

import terminal
import telnetlib
import struct
import sys

_ESC = chr(0x1b)
_SS3 = _ESC + chr(0x4f)
_CSI = _ESC + chr(0x5b)

keyMapSpecial = {
 # normal, shift, ctrl, ...
 KEYIDS["KEY_ESC"]: (_ESC, _ESC),

KEYIDS["KEY_F1"]:  (_SS3 + "P",   _CSI + "1P;%s~"),
 KEYIDS["KEY_F2"]:  (_SS3 + "Q",   _CSI + "1Q;%s~"),
 KEYIDS["KEY_F3"]:  (_SS3 + "R",   _CSI + "1R;%s~"),
 KEYIDS["KEY_F4"]:  (_SS3 + "S",   _CSI + "1S;%s~"),
 KEYIDS["KEY_F5"]:  (_CSI + "15~", _CSI + "15;%s~"),
 KEYIDS["KEY_F6"]:  (_CSI + "17~", _CSI + "17;%s~"),
 KEYIDS["KEY_F7"]:  (_CSI + "18~", _CSI + "18;%s~"),
 KEYIDS["KEY_F8"]:  (_CSI + "19~", _CSI + "19;%s~"),
 KEYIDS["KEY_F9"]:  (_CSI + "20~", _CSI + "20;%s~"),
 KEYIDS["KEY_F10"]: (_CSI + "21~", _CSI + "21;%s~"),
 KEYIDS["KEY_F11"]: (_CSI + "23~", _CSI + "23;%s~"),
 KEYIDS["KEY_F12"]: (_CSI + "24~", _CSI + "24;%s~"),

KEYIDS["KEY_UP"]:    (_CSI + "A", _CSI + "1;%sA"),
 KEYIDS["KEY_DOWN"]:  (_CSI + "B", _CSI + "1;%sB"),
 KEYIDS["KEY_RIGHT"]: (_CSI + "C", _CSI + "1;%sC"),
 KEYIDS["KEY_LEFT"]:  (_CSI + "D", _CSI + "1;%sD"),

KEYIDS["KEY_HOME"]:     (_CSI + "H",  _CSI + "1;%sH"),
 KEYIDS["KEY_END"]:      (_CSI + "F",  _CSI + "1;%sF"),
 KEYIDS["KEY_INSERT"]:   (_CSI + "2~", _CSI + "2;%s~"),
 KEYIDS["KEY_DELETE"]:   (_CSI + "3~", _CSI + "3;%s~"),
 KEYIDS["KEY_PAGEUP"]:   (_CSI + "5~", _CSI + "5;%s~"),
 KEYIDS["KEY_PAGEDOWN"]: (_CSI + "6~", _CSI + "6;%s~")
}

keyMapAscii = {
 # normal, shif, ctrl, unused
 KEYIDS["KEY_1"]: ("1", "!", "", ""),
 KEYIDS["KEY_2"]: ("2", "@", "", ""),
 KEYIDS["KEY_3"]: ("3", "#", chr(0x1b), ""),
 KEYIDS["KEY_4"]: ("4", "$", chr(0xff)+chr(0xf3)+chr(0xff)+chr(0xfd)+chr(0x06), ""),
 KEYIDS["KEY_5"]: ("5", "%", chr(0x1d), ""),
 KEYIDS["KEY_6"]: ("6", "^", chr(0x1e), ""),
 KEYIDS["KEY_7"]: ("7", "&", chr(0x1f), ""),
 KEYIDS["KEY_8"]: ("8", "*", "", ""),
 KEYIDS["KEY_9"]: ("9", "(", "", ""),
 KEYIDS["KEY_0"]: ("0", ")", "", ""),

KEYIDS["KEY_A"]: ("a", "A", chr(0x01), ""),
 KEYIDS["KEY_B"]: ("b", "B", chr(0x02), ""),
 KEYIDS["KEY_C"]: ("c", "C", chr(0x03), ""), # ctrl = chr(0xff)+chr(0xf4)+chr(0xff)+chr(0xfd)+
chr(0x06)
 KEYIDS["KEY_D"]: ("d", "D", chr(0x04), ""),
 KEYIDS["KEY_E"]: ("e", "E", chr(0x05), ""),
```

```
 KEYIDS["KEY_F"]: ("f", "F", chr(0x06), ""),
 KEYIDS["KEY_G"]: ("g", "G", chr(0x07), ""),
 KEYIDS["KEY_H"]: ("h", "H", chr(0x08), ""),
 KEYIDS["KEY_I"]: ("i", "I", chr(0x09), ""),
 KEYIDS["KEY_J"]: ("j", "J", chr(0x0a), ""),
 KEYIDS["KEY_K"]: ("k", "K", chr(0x0b), ""),
 KEYIDS["KEY_L"]: ("l", "L", chr(0x0c), ""),
 KEYIDS["KEY_M"]: ("m", "M", chr(0x0d), ""),
 KEYIDS["KEY_N"]: ("n", "N", chr(0x0e), ""),
 KEYIDS["KEY_O"]: ("o", "O", chr(0x0f), ""), # ctrl = chr(0xff)+chr(0xf5)+chr(0xff)+chr(0xfd)+
chr(0x06)
 KEYIDS["KEY_P"]: ("p", "P", chr(0x10), ""),
 KEYIDS["KEY_Q"]: ("q", "Q", chr(0x11), ""),
 KEYIDS["KEY_R"]: ("r", "R", chr(0x12), ""),
 KEYIDS["KEY_S"]: ("s", "S", chr(0x13), ""),
 KEYIDS["KEY_T"]: ("t", "T", chr(0x14), ""),
 KEYIDS["KEY_U"]: ("u", "U", chr(0x15), ""),
 KEYIDS["KEY_V"]: ("v", "V", chr(0x16), ""),
 KEYIDS["KEY_W"]: ("w", "W", chr(0x17), ""),
 KEYIDS["KEY_X"]: ("x", "X", chr(0x18), ""),
 KEYIDS["KEY_Y"]: ("y", "Y", chr(0x19), ""),
 KEYIDS["KEY_Z"]: ("z", "Z", chr(0x1a), ""), # ctrl = chr(0xff)+chr(0xed)+chr(0xff)+chr(0xfd)+
chr(0x06)

KEYIDS["KEY_ENTER"]: ("\r\n", "\r\n", "\r\n", ""),
 KEYIDS["KEY_TAB"]: ("\t", _CSI + chr(0x5a), "", ""),
 KEYIDS["KEY_SPACE"]: (" ", "", chr(0x00), ""),

KEYIDS["KEY_COMMA"]:      (",", "<", "", ""),
 KEYIDS["KEY_DOT"]:        (".", ">", "", ""),
 KEYIDS["KEY_SLASH"]:      ("/", "?", chr(0x1f), ""),
 KEYIDS["KEY_MINUS"]:      ("-", "_", "", ""),
 KEYIDS["KEY_EQUAL"]:      ("=", "+", "", ""),
 KEYIDS["KEY_LEFTBRACE"]:  ("[", "{", _ESC, ""),
 KEYIDS["KEY_RIGHTBRACE"]: ("]", "}", "", ""),
 KEYIDS["KEY_SEMICOLON"]:  (";", ":", "", ""),
 KEYIDS["KEY_APOSTROPHE"]: ("'", "\"", "", ""),
 KEYIDS["KEY_BACKSLASH"]:  ("\\", "|", chr(0xff)+chr(0xf3)+chr(0xff)+chr(0xfd)+chr(0x06), ""),
 KEYIDS["KEY_GRAVE"]:      ("`", "~", chr(0x31), ""),

KEYIDS["KEY_BACKSPACE"]: ("", "", chr(0x08), ""),
}

def getFlags(isShift, isCtrl, isAlt):
 flags = 0
 if isShift:
  flags |= 1
 if isAlt:
  flags |= 2
 if isCtrl:
  flags |= 4
 return flags

def getKeyXtermString(key, isShift, isCtrl, isAlt):
 if  keyMapSpecial.has_key(key):
  flags = getFlags(isShift, isCtrl, isAlt) + 1
  if flags == 1:
   return keyMapSpecial[key][0]
  else:
   return keyMapSpecial[key][1] % flags
```

```
elif keyMapAscii.has_key(key):
  if isShift:
   return keyMapAscii[key][1]
  elif isCtrl:
   return keyMapAscii[key][2]
  else:
   return keyMapAscii[key][0]
 else:
  return ""

class ConsoleKeyboardInput:
 def __init__(self, console):
  self.console = console
  self.isShift = False
  self.isCtrl  = False

  self.enigmaFixed = False
  # set enigmaFixed to True when enigma is fixed to:
  #   - send keyUp after even if keyUp.code != lastDownKey.code
  #   - not ignore ctrl, shift, etc keys on ascii mode
  if self.enigmaFixed:
   console.setKeyboardModeAscii()
  else:
   console.setKeyboardModeNone()

def execBegin(self):
  eActionMap.getInstance().bindAction("", -20, self.inputKey)

def execEnd(self):
  eActionMap.getInstance().unbindAction("", self.inputKey)

def destroy(self):
  pass

def inputKey(self, key, flag):
  try:
   print "input key=%s flag=%s" % (key, flag)

   if key == KEYIDS["KEY_EXIT"]:
    if flag == 0:
     self.console.cancel()
     return 1
   elif key == KEYIDS["KEY_LEFTSHIFT"] or key == KEYIDS["KEY_RIGHTSHIFT"]:
    if flag == 0:
     print 'Shift DOWN'
     self.isShift = True
    elif flag == 1:
     print 'Shift UP'
     if self.enigmaFixed:
      self.isShift = False
    return 1
   elif key == KEYIDS["KEY_LEFTCTRL"] or key == KEYIDS["KEY_RIGHTCTRL"]:
    if flag == 0:
     print 'Ctrl DOWN'
     self.isCtrl = True
    elif flag == 1:
     print 'Ctrl UP'
     if self.enigmaFixed:
      self.isCtrl = False
    return 1
```

```
        elif key == KEYIDS["KEY_ASCII"]:
          keyStr = str(unichr(getPrevAsciiCode()));
          if self.isCtrl:
            if self.console.inputKey( char(getPrevAsciiCode() & 0x1f) ):
              return 1
          else:
            if self.console.inputKey( str(unichr(getPrevAsciiCode())) ):
              return 1
          return 0
        else:
          print 'Find key string ... (flag=%s)' % (flag)

          if flag != 1:
            return 1

          if self.console.inputKey( getKeyXtermString(key, self.isShift, self.isCtrl, False) ):
            if not self.enigmaFixed:
              self.isShift = False
              self.isCtrl = False
            return 1
          else:
            print 'Input unknown key: %s %s' % (key,(key_name for key_name,value in KEYIDS.items() ⤸
            if value==key).next())
            return 0
      except:
        print 'Input unknown key: ', key
        return 0


class TelnetConsole(Screen):
  def __init__(self, session, width=80, height=20, host="localhost", port=23):
    print "TelnetConsole start with width=%s, height=%s, host=%s" % (width, height, host)

    Screen.__init__(self, session)
    Screen.setTitle(self, "Telnet Console")
    self.skinName = "Console"

    self["text"] = ScrollLabel("")
    self["actions"] = ConsoleKeyboardInput(self)

    try:
      self.term = terminal.Terminal(height, width)
      self.telnetConn = telnetlib.Telnet(host, port)
      nawsCmd = struct.pack('!BBBHHBB', 255, 250, 31, width, height, 255, 240) # IAC SB NAWS
      self.telnetConn.get_socket().send(nawsCmd)
    except:
      errorMsg = "Unable connect to host %s:%d\n\nERROR: %s: %s" % (host, port, sys.exc_info()[0]⤸
      .__name__, sys.exc_info()[1])
      print errorMsg
      self.close(errorMsg)

    self.checkTimer = eTimer()
    self.checkTimer.callback.append(self.checkNewData)
    self.checkTimer.start(300)

    print "TelnetConsole start done"


  def checkNewData(self):
    try:
```

```
    newData = self.telnetConn.read_very_eager()
    if newData != '':
      print "New DATA from telnet: ", newData
      self.term.write(newData)
      self.onUpdateTerm()
  except EOFError: pass


def onUpdateTerm(self):
  results = ""
  (scrollback, screen) = self.term.dump_plain()
  for line in screen:
    for char in line:
      results = results + char
    results = results + "\n"
  self["text"].setText( results.encode('utf-8') )


def inputKey(self, asciiString):
  if asciiString != "":
    try:
      self.telnetConn.write(asciiString)
      return True
    except:
      return False
  else:
    return False


def cancel(self):
  print "TelnetConsole ending"
  self.checkTimer.stop()
  del self.checkTimer
  self.telnetConn.close()
  self.close(None)


class TelnetConsoleConfig(Screen, ConfigListScreen):
 def __init__(self, session, width=80, height=20, host="localhost", port=23):
  Screen.__init__(self, session)
  Screen.setTitle(self, "Telnet Client")
  self.skinName = "Setup"

  self.port = 23
  self.host = "localhost"

  self["actions"] = ActionMap(["TeletextActions"],
  {
   "ok":     self.start,
   "exit":   self.cancel,
   "red":    self.cancel,
   "green":  self.editAndRun,
   "yellow": self.enterHost,
   "blue":   self.enterPort,
  }, -2)
  self["key_red"]    = StaticText("Exit")
  self["key_green"]  = StaticText("Edit and Run")
  self.editAndRunMode = False

  self.list = [ ]
```

```
  ConfigListScreen.__init__(self, self.list, session)
  self.inputHost = ConfigText(default=self.host, fixed_size=False, visible_width=70)
  self.list.append(getConfigListEntry(_("Host:"), self.inputHost))
  self.inputPort = ConfigInteger(default=self.port, limits=(1,65535))
  self.list.append(getConfigListEntry(_("Port:"), self.inputPort))
  self["config"].setList(self.list)


#
 # edit via InputBox
 #
 def editAndRun(self):
  self.editAndRunMode = True;
  self.enterHost()

def enterHost(self):
  self.host = self.inputHost.getText()
  self.session.openWithCallback(self.callbackHost, InputBox, title="Enter host name or IP", ⤸
  text=self.host, windowTitle="Telnet Client")

def callbackHost(self, val):
  self.host=val
  self.inputHost.setValue(self.host);
  if (self.editAndRunMode):
   self.enterPort()

def enterPort(self):
  self.port = self.inputPort.getValue()
  self.session.openWithCallback(self.callbackPort, InputBox, title="Enter port number", ⤸
  text=str(self.port), type=Input.NUMBER, windowTitle="Telnet Client")

def callbackPort(self, val):
  self.port=int(val)
  self.inputPort.setValue(self.port);
  if (self.editAndRunMode):
   self.start()

#
 # run and exit
 #
 def start(self):
  self.host = self.inputHost.getText()
  self.port = self.inputPort.getValue()
  print "start telnet session to %s:%d" % (self.host, self.port)
  if self.inputHost.help_window is not None:
   self.inputHost.help_window.hide()
  self.hide()
  self.session.openWithCallback(self.onTelnetExit, TelnetConsole, host=self.host, port=self.⤸
  port)

def onTelnetExit(self, errorMsg):
  self.show()
  if self.inputHost.help_window is not None:
   self.inputHost.help_window.show()
  if errorMsg is not None:
   self.session.open(MessageBox, errorMsg, MessageBox.TYPE_ERROR)

def cancel(self):
  self.close()
```

```
def main(session, **kwargs):
 session.open(TelnetConsoleConfig)

def Plugins(**kwargs):
 return PluginDescriptor(name="Telnet Client", icon="telnetClient.png", description="Client ⤸
 of telnet protocol", where = PluginDescriptor.WHERE_PLUGINMENU, fnc=main)
```

USERBOUQUET-NAMES.SH

```
#!/bin/bash

# adding chanell names (based on lamedb info) to userbouquet entries for easy userbouquet ⤸
editing as text file

if [ $# -ne 2 ]; then
 echo USAGE $0 /etc/enigma2/lamedb /etc/enigma2/userbouquet.LastScanned.tv
exit
fi

tr -d '\r' < $2 | while read line; do
 id=`echo $line | awk -F: '/^#SERVICE / && NF == 11 {
  printf ("^0*%s:0*%s:0*%s:0*%s\n", $4, $7, $5, $6)
 }'`
if [ "$id" != "" ]; then
  name=`grep -i -A 1 "$id" $1 | tr -d '\r' | tail -n1`
  echo "$line:$name"
else
  echo "$line"
fi
done


# cd /etc/enigma2/
# echo "#NAME Podstawowe (TV)" > userbouquet.Podstawowe.tv
# echo "#SERVICE 1:64:0:0:0:0:0:0:0:0::nowo dodane" >> userbouquet.Podstawowe.tv
# echo "#DESCRIPTION nowo dodane" >> userbouquet.Podstawowe.tv
# a() { grep "$1" userbouquet.HotBird.tv >> userbouquet.Podstawowe.tv; }
# a TVN
```

Listy kanałów

W enigma2 występują dwa główne miejsca konfiguracji list kanałów - są to plik `/etc/enigma2/lamedb` oraz plik `/etc/enigma2/bouquets.tv` (wraz z plikami do niego podłączonymi).

LAMEDB

W pliku tym znajdują się informacje związane z odbiorem poszczególnych stacji - informacje na temat używanych transponderów oraz informacje na temat parametrów poszczególnych programów (transponder i parametry w ramach niego).

Listy parametrów transponderów znajdują się pomiędzy liniami "transponders" i "end". Opis parametrów transpondera ma postać:

```
[NAMESPACE ID]:[TRANSPORT STREAM ID]:[NETWORK ID]
```

```
        [type] [frequency kHz]:[symbol
↪   rate]:[polaization]:[fec]:[position]:[inversion]:[flags]:[system]:[modulation]:[rolloff]:[pilot]
/
```

gdzie:

- type: s => DVB-S* t => DVB-T c => DVB-C
- polaization: 0 => Horizontal, 1 => Vertical, 2 => left, 3 => right
- system: 0 => DVB-S, 1 => DVB-S2
- cztery ostatnie pola (:[system]:[modulation]:[rolloff]:[pilot]) nie występują w części wpisów

Listy parametrów usług / stacji TV (pomiędzy liniami ”services” i ”end”), opis parametrów usługi ma postać:

```
[SERVICE ID]:[NAMESPACE ID]:[TRANSPORT STREAM ID]:[NETWORK ID]:[type 1=TV
↪   2=RADIO]:[number]
[nazwa usługi]
p:[nazwa dostawcy],c:[cached data],C:[card ID],f:[flags]
```

Zobacz także w Sieci: Lamedb format explained[1942].

BOUQUETS.TV

bouquets.tv oraz includowane przez niego (poprzez wpis `#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "nazwa_pliku" ORDER BY bouquet`) pliki zawierają m.in. wpisy typu:

- stacja TV: `#SERVICE 1:0:[service type]:[SERVICE ID]:[TRANSPORT STREAM ID]:[NETWORK ID]:[NAMESPACE ID]:[parent service ID]:[parent transport stream ID]:0::[komentarz]`, gdzie:
  - service type - typowo 1
  - SERVICE ID - identyfikuje usługę na transponderze (zestaw strumieni danej stacji)
  - TRANSPORT STREAM ID + NETWORK ID + NAMESPACE ID - identyfikuje multiplex (transponder, kanał)
  - parent service ID - typowo 0
  - parent transport stream ID - typowo 0
- separator: `#SERVICE 1:64:0:0:0:0:0:0:0:0::sep`
- pominięty numer: `#SERVICE 1:832:0:0:0:0:0:0:0:0::noop`
- nazwa: `#DESCRIPTION [nazwa dla poprzedzającego wpisu #SERVICE]`, dla kanałów nazwa standardowo pobierana jest z lamedb więc nie ma potrzeby stosowania tego pliku)

Pierwsze pole w #SERVICE określa typ usługi:

- 0 => Structure
- 1 => DVB
- 2 => File

Drugie pole w #SERVICE jest flagą binarną:

- 1 => isDirectory
- 2 => mustDescent
- 3 => canDescent
- 8 => shouldSort
- 16 => hasSortKey in data[3]
- 32 => sort key ma wartość 1 zamiast 0
- 64 => isMarker
- 128 => isGroup
- 256 => isNumberedMarker (używany z isMarker)
- 512 => isInvisible

Pola od trzeciego do dziesiątego to kolejne elementy tablicy data[]. Natomiast ostatnie jedenaste pole jest postaci: [URL] [nazwa], URL może zawierać dwukropki.

Przykład - `/etc/enigma2/bouquets.tv`:

```
#NAME Bouquets (TV)
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "bouquet.001-030.Podstawowe.tv" ORDER BY
↪   bouquet
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "bouquet.031-050.Dodatkowe.tv" ORDER BY bouquet
```

```
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "bouquet.051-080.Przyrodnicze.tv" ORDER BY
↪ bouquet
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "bouquet.081-110.Kids.tv" ORDER BY bouquet
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "bouquet.111-130.Informacyjne.tv" ORDER BY
↪ bouquet
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "bouquet.131-160.Dodatkowe2.tv" ORDER BY
↪ bouquet
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "bouquet.161-180.Music.tv" ORDER BY bouquet
#SERVICE 1:7:1:0:0:0:0:0:0:0:FROM BOUQUET "userbouquet.favourites.tv" ORDER BY bouquet
```

Przykład - fragment `/etc/enigma2/bouquet.051-080.Przyrodnicze.tv` (wpis z `noop_52` powoduje przeskoczenie jednego kanału w numeracji):

```
#NAME Przyrodnicze, naukowe itp
#SERVICE 1:64:0:0:0:0:0:0:0:0::przyrodnicze, naukowe itp
#DESCRIPTION przyrodnicze, naukowe itp
#SERVICE 1:0:1:3B66:12C:13E:820000:0:0:0::Animal Planet HD
#SERVICE 1:832:0:0:0:0:0:0:0:0::noop_52
#SERVICE 1:0:1:1C8B:1CE8:71:820000:0:0:0::Nat Geo Wild
```

Uwagi końcowe

Ważne wpisy konfiguracyjne w `/etc/enigma2/settings`:

```
config.misc.remotecontrol_text_support=false
config.usage.shutdownNOK_action=standby
config.usage.boot_action=standby
config.usage.quickzap_bouquet_change=true
config.skin.primary_skin=Utopia_HD/skin.xml
config.misc.epgcachepath=/media/hdd/
config.usage.allowed_timeshift_paths=['/media/hdd/timeshift/']
config.usage.allowed_autorecord_paths=['/media/hdd/movie/']
config.EMC.movie_pathlimit=/media
config.EMC.movie_homepath=/media/hdd/movie
config.EMC.movie_trashcan_path=/media/hdd/movie/.Trash
config.EMC.movie_preview=true
config.OpenWebif.auth=true
config.av.policy_43=letterbox
```

backup miniFS oraz debianowego chroot'a:

```
cd / && echo -e "dev/*\nproc/*\nsys/*" | busybox tar -X - -cf /media/hdd/miniFS.tar bin
↪ dev etc lib media proc sbin sys
cd /debian/ && echo -e "dev/*\nproc/*\nrun/*\nsys/*" | busybox tar -X - -cf
↪ /media/hdd/debian.tar bin dev etc home lib media mnt proc root sbin sys tmp usr var
```

Gdy nasze skrypty startowe zawiodą możemy odpalić linuxa z `/bin/sh` jako init-em. W tym celu należy połączyć się przez port szeregowy (prędkość 115200) i przed startem bootloadera nacisnąć Ctr+C, zobaczymy wtedy:

```
BCM97424B2 CFE v3.41, Endian Mode: Little
Build Date: Sun May 25 23:34:49 CEST 2014 (paul@ini)
Copyright (C) Broadcom Corporation.
*****************************************
Automatic startup canceled via Ctrl-C
*****************************************
[...]
CFE>
```

W uzyskanym shellu należy wpisać: `boot -z -elf nandflash0.kernel: 'ubi.mtd=0 root=ubi0:rootfs rootfstype=ubifs rw bmem=128M@64M bmem=384M@512M memc1=640M init=/bin/sh'` i nacisnąć Enter.

Monitoring

Dział ten gromadzi projekty programistyczne związane z monitoringiem pracy różnych systemów (zarówno informatycznych jak i infrastrukturalnych). Większość projektów rozwijana jest pod kontem monitoringu serwerowni. Bardziej elektroniczne projekty związane z automatyką domową przedstawione są w dziale inteligentny dom.

Ogólne założenia, wskazówki, itd. dotyczące projektowania obiektów z myślą o inteligentnych instalacjach, projektowania samych instalacji, punktów dystrybucyjnych a także całych serwerowni opisane są w instalacje i budynki.

Na wstępie warto też wspomnieć o otwartym wieloplatformowym oprogramowaniu typu SCADA[1943]: SZARP[1944], OpenSCADA[1945] oraz dużej użyteczności (także w zagadnieniach innych niż monitorowanie infrastruktury IT) programów do monitorowania sieci i serwerów typu Zabbix[1946], Nagios[1947], Cacti[1948] i innych[1949].

Zabbix

Zabbix[1950] jest rozbudowanym systemem do monitorowania hostów i stanu sieci, można powiedzieć że łączy funkcjonalność systemów takich jak Nagios[1951] i Munin[1952]. Konfiguracja odbywa się przez interfejs WWW i trzymana jest (podobnie jak dane) w bazie SQL, może być jednak importowana i eksportowana do XMLa co umożliwia jej łatwa masową edycję. Poniżej zamieszczam kilka skryptów i projektów związanych z tym systemem:

- zabbix.c i zabbix.h - prosta implementacja sendera zabbixowego
- get_http_code.sh - skrypt przeznaczony do umieszczenia w `/etc/zabbix/externalscripts` i używania w testach na kod odpowiedzi HTTP
- zabbix_get_from_history.sh - skrypt pobierający ostatnia wartość wskazanego itemu z bazy zabbix'a
- klimy.sh i klimy.xsl - skrypt przeznaczony do odpalania z cron'a i wstawiania do zabbixa danych pobranych z klimatyzacji (dla Emerson Libert HPM)
- make.sh, body.template.xml, header.xml, footer.xml - system generujący szablon do zabbixa w oparciu o plik konfiguracji sql-owej Master modbus RTU
- analizatory.php - skrypt php wyciągający z bazy zabbixowej ostatnie wartości dla wskazanego analizatora PM710 i prezentujący je na WWW

Zobacz też: Comparison of network monitoring systems[1953], Monit[1954]

get_http_code.sh

```
#!/bin/bash

URL=$2

TMP_FILE=`mktemp /tmp/zabbix_curl.XXXXXX`
CODE=`curl -w '%{http_code}' -o $TMP_FILE -s "$URL"`

echo $CODE

FILE="/tmp/zabbix_curl.`echo "$URL" | tr '/:?' '___'`.html"
if [ $CODE -eq 200 ]; then
 mv $TMP_FILE ${FILE}_ok
else
 if [ -f ${FILE}_ok ]; then
  mv $TMP_FILE ${FILE}_`date +"%F_%T"`
  rm -f ${FILE}_ok
else
  mv $TMP_FILE ${FILE}_error
fi
fi
```

KLIMY.SH

```
#!/bin/bash

hostlist="HPM_1 HPM_2"

for h in $hostlist; do
 wget -O - http://${h//_/-} 2>/dev/null |
  xalan -xsl /etc/zabbix/data_fetching_scripts/klimy.xsl -param hostname "'$h'" 2>/dev/null
done | zabbix_sender -z 127.0.0.1 -i - | grep 'Failed [^0]'
```

KLIMY.XSL

```
<?xml version="1.0"?>

<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fn="http://www.w3.org/2005/xpath-functions"
exclude-result-prefixes="fn"
>
 <xsl:output method='text' encoding='utf-8' />
 <xsl:param name="hostname" />

<xsl:variable name="LF"><xsl:text>
</xsl:text></xsl:variable>

<xsl:strip-space elements="*" />

<xsl:template match="Item">
  <xsl:choose>
   <xsl:when test="Label='Return Air Temperature'">
    <xsl:value-of select="$hostname" />
    <text> temperatura </text>
    <xsl:value-of select="Value"/>
    <xsl:value-of select="$LF"/>
   </xsl:when>
   <xsl:when test="Label='Return Air Humidity'">
    <xsl:value-of select="$hostname" />
    <text> wilgotnosc </text>
    <xsl:value-of select="Value"/>
    <xsl:value-of select="$LF"/>
   </xsl:when>
   <xsl:when test="Label='Unit Status'">
    <xsl:value-of select="$hostname" />
    <text> status.unit </text>
    <xsl:apply-templates select="Value" />
    <xsl:value-of select="$LF"/>
   </xsl:when>
   <xsl:when test="Label='Cooling Status'">
    <xsl:value-of select="$hostname" />
    <text> status.cooling </text>
    <xsl:apply-templates select="Value" />
    <xsl:value-of select="$LF"/>
   </xsl:when>
   <xsl:when test="Label='Electrical Heater Status'">
    <xsl:value-of select="$hostname" />
    <text> status.heater </text>
    <xsl:apply-templates select="Value" />
```

```
      <xsl:value-of select="$LF"/>
    </xsl:when>
    <xsl:when test="Label='Dehumdification Status'">
      <xsl:value-of select="$hostname" />
      <text> status.dehumdification </text>
      <xsl:apply-templates select="Value" />
      <xsl:value-of select="$LF"/>
    </xsl:when>
    <xsl:when test="Label='Humidifier Status'">
      <xsl:value-of select="$hostname" />
      <text> status.humidifier </text>
      <xsl:apply-templates select="Value" />
      <xsl:value-of select="$LF"/>
    </xsl:when>
  </xsl:choose>
 </xsl:template>
 <xsl:template match="Value">
  <xsl:choose>
    <xsl:when test="text()='On'">
      <text>1</text>
    </xsl:when>
    <xsl:when test="text()='Unit On'">
      <text>1</text>
    </xsl:when>
    <xsl:when test="text()='Off'">
      <text>0</text>
    </xsl:when>
    <xsl:when test="text()='Unit Off'">
      <text>0</text>
    </xsl:when>
    <xsl:otherwise>
      <text>2</text>
    </xsl:otherwise>
  </xsl:choose>
 </xsl:template>
</xsl:stylesheet>
```

ZABBIX_GET_FROM_HISTORY.SH

```
#!/bin/bash

USER=zabbix
PASSWORD=secret
DATABASE=zabbix

if [ $# -ne 2 ]; then
 echo "USAGE: $0 hostname item.key"
exit
fi

echo "SELECT v.value FROM hosts h JOIN items i JOIN history v ON
 (i.hostid=h.hostid AND v.itemid = i.itemid AND v.clock = i.lastclock AND h.host='$1' AND i.↩
 key_='$2');" |
mysql -u$USER -p$PASSWORD $DATABASE | tail -n1
```

System powiadomień SMS

Projekt stanowi implementację systemu powiadomień SMS dla platformy NPE[1955]. Jako że stanowi dość typowe linuxowe oprogramowanie stosunkowo łatwo może być przeniesiony na dowolna inna platformę zgodną z POSIX.

Obsługa kolejki SMSów (realizowanej jako baza danych SQL)

- sms_queued.c - program wysyłający SMSy w oparciu o wpisy z kolejki
- sms_queue_add.c - program odpowiedzialny za wstawianie SMSów do kolejki

System odbierający e-maile i tworzący w oparciu o nie wpisy w kolejce SMSów:

- smtp_listener.c - daemon nasłuchujący połączeń SMTP
- smtp_talk.c i smtp_talk.h - biblioteczka obsługująca dialog SMTP (wywołuje mail2sms.sh oraz wstawia maila do kolejki nullmailer'a)
- mail2sms.sh - skrypt parsujący e-mail'e i decydujący o dodaniu wiadomości do kolejki SMSów

Pliki ogólne:

- error_reporting.h - plik nagłówkowy definiujący makra odpowiedzialne za wyświetlanie bądź zapisywanie do logu systemowego informacji zgłaszanych przez program
- install_and_run.sh[1956] - skrypt używany do zainstalowania i uruchomienia systemu na platformie NPE (podpinany do skryptu startowego systemu)
- Makefile[1957]
- nullmailer.build.sh[1958] - skrypt obudowujący kompilację nullmailer'a[1959] dla NPE

Pobierz[1960]

Koncepcje dalszego rozwoju

- Można rozważyć mechanizm limitowania ilości wysłanych SMSów na jeden numer danym czasie z uwględnieniem priorytetów poprzez procedurę:
  1. `DELETE FROM log WHERE phone= AND TIME < curr-0.5h`
  2. `num = SELECT count(*) FROM log WHERE phone=`
  3. `if (num>4 and priority > 30) drop  else if (num>8 and priority > 10) drop else send`

INSTALL_AND_RUN.SH

```sh
#!/bin/sh

exec 1>/var/log/smtp2sms.log
exec 2>&1

BASEDIR='/mnt/nand'

#
# prepare system
#
chmod a+x /usr/bin/gawk

if ! grep "^mail:x:8" /etc/passwd > /dev/null; then
 echo "mail:x:8:99:Mail2SMS system:/tmp:/bin/sh" >> /etc/passwd
fi

ln -s $BASEDIR/smtp2sms /opt/

#
# run nullmailer
#
mkdir -p '/usr/lib/nullmailer/'
ln -s /opt/smtp2sms/nullmailer/lib/* '/usr/lib/nullmailer/'
ln -s /opt/smtp2sms/nullmailer/bin/* '/usr/bin/'
ln -s '/opt/smtp2sms/nullmailer/lib/libstdc++.so.6.0.8'  '/usr/lib/libstdc++.so.6'
mkdir /etc/nullmailer
```

```
echo "192.168.119.13" > /etc/nullmailer/remotes

mkdir -p /var/spool/nullmailer/queue
mkfifo /var/spool/nullmailer/trigger

chown -R mail /var/spool/nullmailer

su mail -c "exec /opt/smtp2sms/nullmailer/bin/nullmailer-send" 2>&1 | logger -t nullmailer &

#
# run smtp_listener
#
touch /var/log/mail2sms.log
chown mail /var/log/mail2sms.log
/opt/smtp2sms/smtp_listener &

#
# run SMS QUEUE
#
chown mail /dev/ttyS1
chmod a+w /tmp/
#su mail -c "exec /opt/smtp2sms/sms_queued 8459" &
su mail -c "exec /opt/smtp2sms/sms_queued" &
```

MAIL2SMS.SH

```
#!/bin/sh

/usr/bin/gawk -F': *' -v from_email=$1 -v to_email=$2 '
 function send_sms(sms_priority, sms_number, sms_message) {
  cmd = sprintf("/opt/smtp2sms/sms_queue_add %s %d \"%s\"", sms_priority, sms_number, ⤸
  sms_message)
  printf("RUN: %s\n", cmd);
  system(cmd)
 }

 BEGIN {
  MAIL_ENTRY_HEAD = 0
  MAIL_ENTRY_BODY = 1
  MAIL_ENTRY_END = -1

  mail_entry = MAIL_ENTRY_HEAD

  MESSAGE_TYPE_UNKNOW = 0
  MESSAGE_TYPE_GENERIC = 1
  MESSAGE_TYPE_BMS = 2
  MESSAGE_TYPE_ZABBIX = 3

  message_type = MESSAGE_TYPE_UNKNOW

  message = ""
  priority = ""
  phone = ""
 }

 mail_entry == MAIL_ENTRY_END {
  next;
 }
```

```
#                     #
#  PARSE MAIL HEADER  #
#                     #

mail_entry == MAIL_ENTRY_HEAD && $1 == "X-SMS-Number" {
 phone = $2;
}

mail_entry == MAIL_ENTRY_HEAD && $1 == "X-SMS-Priority" {
 priority = $2;
}

mail_entry == MAIL_ENTRY_HEAD && $1 == "Subject" {
 header_subject = $2
 if ($2 ~ "Alarm z BMS")
  message_type = MESSAGE_TYPE_BMS;
}

mail_entry == MAIL_ENTRY_HEAD && $1 == "From" {
 header_from = $2
 if ($2 ~ "zabbix@icm.edu.pl")
  message_type = MESSAGE_TYPE_ZABBIX;
}

mail_entry == MAIL_ENTRY_HEAD {
 # check for end of headers
 if ($0 ~ /^\r*$/) {
  mail_entry = MAIL_ENTRY_BODY;
  if (message_type == MESSAGE_TYPE_UNKNOW && phone != "" && priority != "")
   message_type = MESSAGE_TYPE_GENERIC
 }
 # do not parse body if in header
 next;
}


#                     #
#   PARSE MAIL BODY   #
#                     #

message_type == MESSAGE_TYPE_GENERIC {
 message = $0
}

message_type == MESSAGE_TYPE_BMS && $1 == "Site number" {
 gsub("^.*Priority", "", $2)
 gsub("\r*$", "", $2)
 priority = $2
}
message_type == MESSAGE_TYPE_BMS && $1 ~ "Alarm block description" {
 gsub("^ *", "", $2)
 gsub("\r*$", "", $2)
 message = $2
}

message_type == MESSAGE_TYPE_ZABBIX && $1 ~ "Problem$" {
 msg1 = $2
}
```

```
 message_type == MESSAGE_TYPE_ZABBIX && $1 ~ "Problem status$" {
  message = sprintf("%s: %s", msg1, $2)
 }


 #                      #
 #   PUT SMS TO QUEUE   #
 #                      #

 phone != "" &&  message != "" && priority != "" {
  # wiadomosci na wskazany numer
  send_sms(priority, phone, message);
  mail_entry = MAIL_ENTRY_END;
  next
 }

 message != "" && priority != "" && message_type == MESSAGE_TYPE_BMS {
  # wiadomosci z systemu BMS - adresaci ustalani w oparciu o priorytet
  if (priority < 40 && priority != 6) {
   send_sms(priority, 12345, message);
   send_sms(priority, 22345, message);
   send_sms(priority, 32345, message);
   mail_entry = MAIL_ENTRY_END;
  } else if (priority == 99) {
   message = sprintf("TRAFO: %s", message)
   send_sms(40, 45678, message);
   mail_entry = MAIL_ENTRY_END;
  }
 }
 END {printf ("END %s - %s - %s\n", priority, phone, message);}
' | logger -p local0.info -t mail2sms
```

SMS_QUEUE_ADD.C

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <termios.h>
#include <sqlite3.h>

#define NO_USE_SYSLOG
#include "error_reporting.h"

#define DATABESE_FILE "/tmp/sms_spool.db"

int main(int argc, char **argv) {
  if (argc != 4) {
   LOG_PRINT_CRIT("USAGE: %s PRIORITY PHONE MESSAGE", argv[0]);
   exit(-1);
  }

// open or create database
 sqlite3 *db;
 if ( sqlite3_open(DATABESE_FILE, &db) ) {
  LOG_PRINT_CRIT("Can't open database: %s", sqlite3_errmsg(db));
  sqlite3_close(db);
```

```
  exit(-1);
 }

char buf[255];
 snprintf(buf, 255, "INSERT INTO queue (time, pri, tel, msg) VALUES (%d, %s, %s, '%s');",
  (int)time(NULL), argv[1], argv[2], argv[3]);

// putting SMS to queue
 sqlite3_stmt *stmt;
 int ret;
 if ( ( ret = sqlite3_prepare_v2(db, buf, -1, &stmt, NULL) ) != SQLITE_OK ) {
  LOG_PRINT_CRIT("Can't add SMS to queue - sqlite3_prepare_v2() exit with code %d", ret);
  sqlite3_close(db);
  exit(-1);
 }

while ( ( ret = sqlite3_step(stmt) ) == SQLITE_BUSY ) {
  LOG_PRINT_WARN("Waiting for delete SMS from queue");
  usleep( 1000 );
 }
 if ( ret != SQLITE_DONE ) {
  LOG_PRINT_CRIT("Can't add SMS to queue - sqlite3_step() exit with code %d", ret);
  sqlite3_close(db);
  exit(-1);
 }

if ( sqlite3_finalize(stmt) ) {
  LOG_PRINT_CRIT("Adding SMS to queue ERROR - sqlite3_step() exit with code %d", ret);
  sqlite3_close(db);
  exit(-1);
 }

// clean up
 sqlite3_close(db);
 return 0;
}
```

SMS_QUEUED.C

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <termios.h>
#include <sqlite3.h>

#include "error_reporting.h"

#define DATABESE_FILE "/tmp/sms_spool.db"
#define TTY_DEVICE "/dev/ttyS1"
#define MAX_RETRY_IF_ERROR 4
#define RESTART_MODEM_AFTER_RETRY 2
#define RESTART_MODEM_CMD "/bin/gsmoff; sleep 2; /bin/gsmon; sleep 2"
#define MODEM_TIMEOUT 3
#define MAX_ERROR_DROP 6

int write_to_modem(int tty_fd, const char *txt, const char *expected_return, int sleep_time) {
```

```
 char buf[100];
 int ret, len;
 struct timeval timeout;
 fd_set modem_fd_set;

timeout.tv_sec = 3;
 timeout.tv_usec = 0;

FD_ZERO(&modem_fd_set);
 FD_SET(tty_fd, &modem_fd_set);

len=strlen(txt);
 ret = write(tty_fd, txt, len);
 if (ret != len) {
  LOG_PRINT_WARN("Error write data to modem: %m");
  return -1;
 }

sleep(sleep_time);

ret = select(tty_fd+1, &modem_fd_set, NULL, NULL, &timeout);
 if (ret > 0) {
  ret = read(tty_fd, buf, 100);
  if (ret < 1) {
   LOG_PRINT_WARN("Error read data from modem - read: %m");
   return -2;
  } else {
   buf[ret-1] = 0;
   LOG_PRINT_INFO(">> %s", buf);
   if (expected_return != NULL && strstr(buf, expected_return) == 0)
    return -4;
   else
    return 0;
  }
 } else if (ret == 0) {
  LOG_PRINT_WARN("Timeout on write to modem");
  return -1;
 } else {
  LOG_PRINT_WARN("Error read data from modem - select: %m");
  return -3;
 }
}

int init_tty() {
 int tty_dev;
 struct termios term;

tty_dev = open (TTY_DEVICE, O_RDWR);
 if (tty_dev < 0) {
  LOG_PRINT_WARN("Opening serial port: %m");
 } else {
  tcgetattr (tty_dev, &term);
  term.c_iflag = IGNBRK | IGNPAR;
  term.c_oflag = term.c_lflag = term.c_line = 0;
  term.c_cflag = CS8 | CREAD |  CLOCAL | HUPCL | B115200;
  tcsetattr (tty_dev, TCSAFLUSH, &term);
 }

return tty_dev;
}
```

```c
int init_modem(int argc, char **argv) {
#ifdef RESTART_MODEM_CMD
 system(RESTART_MODEM_CMD);
#endif

int tty_fd = init_tty();
 if (tty_fd < 0)
  return tty_fd;

if (argc == 2){
  char buf[20];
  snprintf(buf, 20, "at+cpin=%s\r", argv[1]);
  if (write_to_modem(tty_fd, buf, "OK", 2) < 0) {
   LOG_PRINT_WARN("Error sending PIN to modem");
   return -1;
  }
 }

return tty_fd;
}

int send_sms(int tty_fd, const char *to, const char *msg) {
 int ret;
 char buf[100];
 ret = write_to_modem(tty_fd, "ATV1Q0E1\r", "OK", 2);
 if (ret != 0) return ret;
 ret = write_to_modem(tty_fd, "AT+CSCA=\"+48501200777\"\r", "OK", 2);
 if (ret != 0) return ret;
 ret = write_to_modem(tty_fd, "AT+CMGF=1\r", "OK", 2);
 if (ret != 0) return ret;

snprintf(buf, 100, "AT+CMGS=\"+%s\"\r", to);
 ret = write_to_modem(tty_fd, buf, ">", 2);
 if (ret != 0) return ret;

snprintf(buf, 100, "%.97s%c", msg, 0x1a);
 ret = write_to_modem(tty_fd, buf, "OK", 10);
 return ret;
}

int main(int argc, char **argv) {
 openlog("sms_queued", LOG_PID, LOG_DAEMON);

// initialize modem
 int tty_fd = init_modem(argc, argv);
 if (tty_fd < 0) {
  LOG_PRINT_CRIT("Init modem ERROR");
  exit(EXIT_FAILURE);
 }

// database exists ?
 FILE *db_file;
 char init_required = 0;
 db_file=fopen(DATABESE_FILE, "r");
 if (db_file != NULL)
  fclose(db_file);
 else
  init_required = 1;
```

```
// open or create database
 sqlite3 *db;
 if ( sqlite3_open(DATABESE_FILE, &db) ) {
  LOG_PRINT_CRIT("Can't open database: %s", sqlite3_errmsg(db));
  sqlite3_close(db);
  exit(EXIT_FAILURE);
 }

// creating table for queue
 char *ErrMsg = NULL;
 if (init_required == 1) {
  LOG_PRINT_INFO("Initialize database: %s", DATABESE_FILE);
  if ( sqlite3_exec(db, "CREATE TABLE queue (id INTEGER PRIMARY KEY, time INT, pri INT, tel ⤸
  INT, msg TEX);",
    NULL, NULL, &ErrMsg) != SQLITE_OK ) {
   LOG_PRINT_CRIT("Initialize database - SQL error: %s", ErrMsg);
   sqlite3_free(ErrMsg);
   sqlite3_close(db);
   exit(EXIT_FAILURE);
  }
 }

// message queue loop
 int drop_cnt = 0, retry_cnt = 0, last_id = -1, ret_code = 0;
 while (1) {
  sqlite3_stmt *stmt;
  int ret;

  // getting message to send
  if ( ( ret = sqlite3_prepare_v2(db, "SELECT id, time, pri, tel, msg FROM queue ORDER BY pri,⤸
  time LIMIT 1;", -1,
    &stmt, NULL) ) != SQLITE_OK ) {
   LOG_PRINT_ERROR("Getting SMS from queue ERROR - sqlite3_prepare_v2() exit with code %d", ⤸
   ret);
   sqlite3_finalize(stmt);
   continue;
  }
  while ( ( ret = sqlite3_step(stmt) ) == SQLITE_BUSY ) {
   LOG_PRINT_WARN("Waiting for getting SMS from queue");
   usleep( 1000 );
  }

  int id = -3, time = 0, pri = 0;
  const char *text_tmp = NULL;
  char phone_number[16], message[100];
  if ( ret == SQLITE_ROW ) {
   id   = sqlite3_column_int(stmt, 0);
   time = sqlite3_column_int(stmt, 1);
   pri  = sqlite3_column_int(stmt, 2);

   text_tmp = (const char*)sqlite3_column_text(stmt, 3);
   strncpy(phone_number, text_tmp, 16);
   text_tmp = (const char*)sqlite3_column_text(stmt, 4);
   strncpy(message, text_tmp, 100);
  } else if ( ret == SQLITE_DONE ) {
   id = -1;
   usleep( 200000 );
  } else {
   id = -2;
   LOG_PRINT_ERROR("Getting SMS from queue ERROR - sqlite3_step() exit with code %d", ret);
```

```
}

if ( ( ret = sqlite3_finalize(stmt) ) != SQLITE_OK ) {
 LOG_PRINT_ERROR("Getting SMS from queue ERROR - sqlite3_finalize() exit with code %d", ret)⤸
 ;
 continue;
}

if (id < 0)
 continue;

// sending SMS
if (last_id != id) {
 last_id = id;
 retry_cnt=0;
}
LOG_PRINT_INFO("=== SEND SMS (retry %d): id=%d time=%d pri=%d tel=+%s msg=%s ===",
               retry_cnt, id, time, pri, phone_number, message);
ret = send_sms(tty_fd, phone_number, message);
if (ret == 0) {
 LOG_PRINT_INFO("DONE! (id=%d)", id);
 retry_cnt=0;
 drop_cnt=0;
} else {
 LOG_PRINT_INFO("ERROR! (id=%d, code=%d)", id, ret);

 if (retry_cnt < MAX_RETRY_IF_ERROR) {
  if (retry_cnt == RESTART_MODEM_AFTER_RETRY) {
   LOG_PRINT_ERROR("RESTARTING MODEM");
   close(tty_fd);

   tty_fd = init_modem(argc, argv);
   if (tty_fd < 0) {
    LOG_PRINT_CRIT("Reinit modem ERROR");
    exit(EXIT_FAILURE);
   }
  }
  sleep(1);
  retry_cnt++;
  continue;
 } else {
  LOG_PRINT_CRIT("MAX_RETRY_IF_ERROR exceeded - droping this message and sleep for 60s");
  sleep(60);
  drop_cnt++;
  retry_cnt=0;
 }
}

if (drop_cnt > MAX_ERROR_DROP) {
 LOG_PRINT_CRIT("MAX_ERROR_DROP exceeded - exit");
 ret_code = -1;
 break;
}

// remove SMS from queue
if ( ( ret = sqlite3_prepare_v2(db, "DELETE FROM queue WHERE id = ?1;", -1, &stmt, NULL) ) ⤸
!= SQLITE_OK ) {
 LOG_PRINT_WARN("Deleting SMS from queue ERROR - sqlite3_prepare_v2() exit with code %d", ⤸
 ret);
 sqlite3_finalize(stmt);
```

```
      continue;
    }
    if ( ( ret = sqlite3_bind_int(stmt, 1, id) ) != SQLITE_OK ) {
      LOG_PRINT_WARN("Deleting SMS from queue ERROR - sqlite3_bind_int() exit with code %d", ret)⤸
      ;
      sqlite3_finalize(stmt);
      continue;
    }
    while ( ( ret = sqlite3_step(stmt) ) == SQLITE_BUSY ) {
      LOG_PRINT_WARN("Waiting for delete SMS (id=%s) from queue", argv[0]);
      usleep( 1000 );
    }
    if ( ret != SQLITE_DONE ) {
      LOG_PRINT_WARN("Can't delete SMS (id=%s) from queue - sqlite3_step() exit with code %d", ⤸
      argv[0], ret);
      sqlite3_finalize(stmt);
      continue;
    }
    if ( sqlite3_finalize(stmt) ) {
      LOG_PRINT_WARN("Deleting SMS from queue ERROR - sqlite3_step() exit with code %d", ret);
      continue;
    }
  }

// clean up
  sqlite3_close(db);
  close(tty_fd);
  closelog();
  return ret_code;
}
```

SMTP_LISTENER.C

```
// port na którym słuchamy, UID z jakim pracujemy po jego otworzeniu oraz długość ⤸
kolejki
#define PORT 25
#define WORK_AS_UID 8
#define QUERY_SIZE 3

#include <sys/socket.h>
#include <netinet/ip.h>

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>

#include <signal.h>
#include <sys/wait.h>

#include "error_reporting.h"
#include "smtp_talk.h"

int sh;

void end_of_main(int sig) {
  close(sh);
  closelog();
```

```
 exit(0);
}

int main(int argv, char *args[]) {
 openlog("smtp_listener", LOG_PID, LOG_DAEMON);

// gniazdo ...
#ifndef IPv4ONLY
 sh = socket(PF_INET6, SOCK_STREAM, 0);
#else
 sh = socket(PF_INET, SOCK_STREAM, 0);
#endif
 if (sh<0) {
  LOG_PRINT_CRIT("socket: %m");
  exit(EXIT_FAILURE);
 }

// funkcja wykonywana po zakonczeniu potomka - unikamy zoombie
 signal(SIGCHLD, SIG_IGN);

// funkcja wykonywana po Ctrl+C i kill ... unikamy blokowania portu
 signal(SIGTERM, &end_of_main);
 signal(SIGINT, &end_of_main);

// ustawiamy REUSEADDR na sockecie
 int tmp = 1;
 setsockopt(sh, SOL_SOCKET, SO_REUSEADDR, &tmp, sizeof(tmp));

// utworzenie struktury opisującej adres
#ifndef IPv4ONLY
 struct sockaddr_in6 serwer;
 serwer.sin6_family=AF_INET6;
 serwer.sin6_port=htons(PORT);
 serwer.sin6_addr=in6addr_any;

if (bind(sh, (struct sockaddr *) &serwer, sizeof(struct sockaddr_in6)) < 0) {
#else
 struct sockaddr_in serwer;
 serwer.sin_family=AF_INET;
 serwer.sin_port=htons(PORT);
  // host to network short (porzadek bajtow) ... sa tez dla long i odwrotne ...
 serwer.sin_addr.s_addr=INADDR_ANY;
  // INADDR_ANY oznacza że nasłuchujemy na każdym adresie IP danego hosta
  // (zamiast tego można określić konkretny adres)

// przypisanie adresu ...
 if (bind(sh, (struct sockaddr *) &serwer, sizeof(struct sockaddr_in)) < 0) {
#endif
  LOG_PRINT_CRIT("bind: %m");
  exit(EXIT_FAILURE);
 }

// zmiana UIDu
#ifdef WORK_AS_UID
  if (setuid(WORK_AS_UID) < 0) {
  LOG_PRINT_CRIT("Can't drop root privileges: %m");
  close(sh);
  exit(EXIT_FAILURE);
 }
#endif
```

```
// nazwa hosta sie nam przyda ...
 char hostname[64];
 gethostname(hostname, 64);

while(1) {
  // otwarcie portu do nasluchiwania
  if (listen(sh, QUERY_SIZE) < 0) {
   LOG_PRINT_CRIT("listen: %m");
   exit(EXIT_FAILURE);
  }

  // odebranie połączenia
#ifndef IPv4ONLY
  struct sockaddr_in6 from;
  socklen_t fromlen=sizeof(struct sockaddr_in6);
#else
  struct sockaddr_in from;
  socklen_t fromlen=sizeof(struct sockaddr_in);
#endif
  int sh2 = accept(sh, (struct sockaddr *) &from, &fromlen);

  switch (fork()) {
   case -1: // funkcja zwróciła -1 co oznacza błąd
    LOG_PRINT_ERROR("Error in fork");
    break;

   case 0: // funkcja zwróciła zero co oznacza że jesteśmy w procesie potomnym
    close(sh);
    return smtp_talk(sh2, hostname, (struct sockaddr *) &from);

   default: // funkcja zwróciła coś innego od 0 i -1 oznacza to że jestesmy w pocesie ↩
   macierzystym
    close(sh2);
  }
 }

// zamkniecie gniazda
 close(sh);
 closelog();
}
```

SMTP_TALK.C

```
#define SOFTNAME "SMTP server by RRP"
#define MY_EMAIL_ADDRESS "alert@sms-sender"

// uruchomienie polecenia do ktorego zostanie przekazana tresc wiadomosci jako STDIN
// adres nadawcy jako $1 adres odbiorcy jako $2
#define ACTION "/opt/smtp2sms/mail2sms.sh %s %s"

// kolejka dla nullmailer'a ... ten soft powinien chodzic jako ten sam user co nullmailer
#define QUEUE_DIR "/var/spool/nullmailer/queue/"
#define QUEUE_PUSH "/var/spool/nullmailer/trigger"

#define BUF_LINE_SIZE 255
#define BUF_DATA_SIZE 255
```

```
#define _LEN_(s)   ((sizeof(s) - 1) / sizeof(char))
#define _LOG_ stderr

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>

#include <string.h>
#include <ctype.h>

#include <sys/socket.h>
#include <arpa/inet.h>

#include <time.h>

#include "error_reporting.h"
//#include "timeout_exec.h"

void copy_and_clear(char * out, char * in, int len) {
 char *tmp;

tmp = in;
 while (isspace(tmp[0])) tmp++;

strncpy(out, tmp, len);

tmp = strpbrk(out, "\r\n");
 if (tmp) {
  *tmp='\0';
 }
}

char* clear_email(char* in) {
 int len;

len = strlen(in);
 if (in[len-1] == '>')
  in[len-1] = 0;

if (in[0] == '<')
  return in+1;

return in;
}

int read_line(int fd, char *buf, int len) {
 int i, ret;

len = len-1;
 for (i=0; i<len; i++) {
  ret = read(fd, buf+i, 1);
  if (ret < 1) {
   buf[i] = 0;
   return i;
  }
  if (buf[i] == '\n') {
   if (buf[i-1] == '\r') {
    buf[i-1]='\n';
    buf[i] = 0;
```

```
      return i;
    }
    buf[++i] = 0;
    return i;
  }
 }

buf[len] = 0;
 return len;
}

int smtp_talk(int sh2, char *hostname, struct sockaddr *ipfrom) {
 char time_str[64];
 char buf[BUF_LINE_SIZE];
 char ip[42], hello[BUF_DATA_SIZE], from[BUF_DATA_SIZE], to[BUF_DATA_SIZE];
 char *from2, *to2;

FILE *cmd_pipe = NULL, *queue = NULL;
 int queue_d = -1;
 int state = 0;
 int msq_num = 0;
 pid_t pid;
 int ret, len;

pid=getpid();

#ifndef IPv4ONLY
 if ( ! inet_ntop(AF_INET6, &(((struct sockaddr_in6*)ipfrom)->sin6_addr), ip, 42) )
#else
 if ( ! inet_ntop(AF_INET, &(((struct sockaddr_in*)ipfrom)->sin_addr), ip, 42) )
#endif
  strcpy(ip, "????");

time_t time_unix = time(0);
 struct tm time_info;
 gmtime_r( &time_unix, &time_info);
 strftime(time_str, 64, "%F %T", &time_info);
 LOG_PRINT_INFO("[%d] connection from: %s at %s", pid, ip, time_str);

strftime(time_str, 64, "%a, %d %b %Y %H:%M:%S %z", &time_info);
 len = snprintf(buf, BUF_LINE_SIZE, "220 %s %s %s\r\n", hostname, SOFTNAME, time_str);
 write(sh2, buf, len);

struct timeval net_timeout;
 fd_set net_fd_set;

while( 1 ) {
  net_timeout.tv_sec = 60;
  net_timeout.tv_usec = 0;

  FD_ZERO(&net_fd_set);
  FD_SET(sh2, &net_fd_set);

  ret = select(sh2+1, &net_fd_set, NULL, NULL, &net_timeout);
  if (ret > 0) {
   if ( read_line(sh2, buf, BUF_LINE_SIZE) == 0 ) {
    break;
   }
  } else {
   break;
```

```
  }

  /// transmited data
  if (state == 4) {
   if (buf[0]=='.' && (buf[1]=='\r' || buf[1]=='\n')) {
    state = 0;
#ifdef ACTION
    if (cmd_pipe) {
     int tmp;
     if ( (tmp = pclose(cmd_pipe)) )
      LOG_PRINT_WARN("[%d.%d] delivered to command ERROR (%d)", pid, msq_num, tmp);
     else
      LOG_PRINT_INFO("[%d.%d] delivered to command OK", pid, msq_num);
     cmd_pipe = NULL;
    }
#endif
#ifdef QUEUE_DIR
    if (queue) {
     int tmp;
     if ( (tmp = fclose(queue)) )
      LOG_PRINT_WARN("[%d.%d] delivered to queue ERROR (%d)", pid, msq_num, tmp);
     else
      LOG_PRINT_INFO("[%d.%d] delivered to queue OK", pid, msq_num);
     queue = NULL;
     queue_d = -1;
# ifdef QUEUE_PUSH
     queue_d = open(QUEUE_PUSH, O_WRONLY | O_NONBLOCK);
     write(queue_d, " ", 1);
     close(queue_d);
     queue_d = -1;
//       wait_for_timeouted_child_exec(2, 0, "exec echo > /var/spool/nullmailer/trigger");
# endif
    }
#endif
    LOG_PRINT_INFO("[%d.%d] end mail", pid, msq_num);
    write(sh2, "250 OK\r\n", _LEN_("250 OK\r\n"));
    msq_num ++;
   } else {
    if (cmd_pipe)
     fputs(buf, cmd_pipe);
    if (queue)
     fputs(buf, queue);
    continue;
   }

  /// start data
  } else if ( strncasecmp(buf, "DATA", _LEN_("DATA")) == 0 ) {
   if (state != 3) {
    write(sh2, "503 valid RCPT command must precede DATA\r\n",
     _LEN_("503 valid RCPT command must precede DATA\r\n"));
   } else {
    from2 = clear_email(from);
    to2 = clear_email(to);
#ifdef ACTION
    char cmd[BUF_DATA_SIZE*2 + _LEN_(ACTION) + 2];
    sprintf(cmd, ACTION, from2, to2);
    cmd_pipe =  popen( cmd, "w" );
#endif
#ifdef QUEUE_DIR
    if ( strcmp(to2, MY_EMAIL_ADDRESS) != 0 ) {
```

```
      strncpy(buf, QUEUE_DIR "/XXXXXX", BUF_LINE_SIZE);
      queue_d = mkstemp(buf);
      if (queue_d < 0) {
       write(sh2, "451 Requested action delayed - Local problem\r\n",
        _LEN_("451 Requested action delayed - Local problem\r\n"));
       continue;
      }
      queue = fdopen(queue_d, "w+");
      if (!queue) {
       write(sh2, "451 Requested action delayed - Local problem\r\n",
        _LEN_("451 Requested action delayed - Local problem\r\n"));
       continue;
      }


      fprintf(queue, "%s\n%s\n\n", from2, to2);

      // "Received:" header
      fprintf(queue, "Received: from [%s] (helo=%s)\n", ip, hello);
      fprintf(queue, "\tby %s with %s\n", hostname, SOFTNAME);
      fprintf(queue, "\t(envelope-from <%s>)\n", from2);
      fprintf(queue, "\tfor %s; %s\n", to2, time_str);
     }
#endif
     write(sh2, "354 Enter message, ending with \".\" on a line by itself\r\n",
      _LEN_("354 Enter message, ending with \".\" on a line by itself\r\n"));
     state = 4;
    }

   /// helo
   } else if ( strncasecmp(buf, "HELO", _LEN_("HELO")) == 0 || strncasecmp(buf, "EHLO", _LEN_(
   "EHLO")) == 0 ) {
    copy_and_clear(hello, buf + _LEN_("HELO") + 1, BUF_DATA_SIZE);
    LOG_PRINT_INFO("[%d.%d] hello: %s", pid, msq_num, hello);
    len = snprintf(buf, BUF_LINE_SIZE, "250 %s Hello %s, pleased to meet you\r\n", hostname, 
    hello);
    write(sh2, buf, len);
    state = 1;

   /// mail form:
   } else if ( strncasecmp(buf, "MAIL FROM:", _LEN_("MAIL FROM:")) == 0 ) {
    copy_and_clear(from, buf + _LEN_("MAIL FROM:"), BUF_DATA_SIZE);
    LOG_PRINT_INFO("[%d.%d] mail from: %s", pid, msq_num, from);
    write(sh2, "250 OK\r\n", _LEN_("250 OK\r\n"));
    state = 2;

   /// rcpt to:
   } else if ( strncasecmp(buf, "RCPT TO:", _LEN_("RCPT TO:")) == 0 ) {
    if (state == 3) {
     write(sh2, "451 Deferred - please try this recipient again later\r\n",
           _LEN_("451 Deferred - please try this recipient again later\r\n"));
    } else if (state < 2) {
     write(sh2, "503 sender not yet given\r\n", _LEN_("503 sender not yet given\r\n"));
    } else {
     copy_and_clear(to, buf + _LEN_("RCPT TO:"), BUF_DATA_SIZE);
     LOG_PRINT_INFO("[%d.%d] mail to: %s", pid, msq_num, to);
     write(sh2, "250 OK\r\n", _LEN_("250 OK\r\n"));
     state = 3;
    }
```

```
  /// quit
  } else if ( strncasecmp(buf, "QUIT", _LEN_("QUIT")) == 0 ) {
   len = snprintf(buf, BUF_LINE_SIZE, "221 %s closing connection\r\n", hostname);
   write(sh2, buf, len);
   break;

  /// reset
  } else if ( strncasecmp(buf, "RSET", _LEN_("RSET")) == 0 ) {
   to[0]=0;
   from[0]=0;
   LOG_PRINT_INFO("[%d.%d] reset", pid, msq_num);
   write(sh2, "250 OK\r\n", _LEN_("250 OK\r\n"));
   state = 0;

  } else {
   write(sh2, "500 unrecognized command\r\n",
    _LEN_("500 unrecognized command\r\n"));
  }
 }

if (ret == 0) {
  LOG_PRINT_WARN("[%d.%d] timeout", pid, msq_num);
  len = snprintf(buf, BUF_LINE_SIZE, "421 %s SMTP command timeout - closing connection\r\n", ⤸
  hostname);
  write(sh2, buf, len);
 }

close(sh2);
 return 0;
}
```

SMTP_TALK.H

```
#ifndef SMTP_TALK_H
#define SMTP_TALK_H

void copy_and_clear(char * out, char * in, int len);
int smtp_talk(int sh2, char *hostname, struct sockaddr *ipfrom);

#endif
```

Master modbus RTU

Projekt stanowi implementację mastera modbus RTU dla platformy NPE[1961]. Jako że stanowi dość typowe linuxowe oprogramowanie stosunkowo łatwo może być przeniesiony na dowolna inna platformę zgodną z POSIX. System umożliwia zapis danych do lokalnej bazy danych opartej na sqlite3[1962] (przy dużej liczbie obsługiwanych rejestrów UPDATE na takiej bazie na platformie NPE jest zbyt mało wydajny) a także zdalny zapis do systemu zabbix (własna implementacja sendera).

pliki źródłowe daemona odpytującego urządzenia poprzez modbus:

— modbus_reader.c - implementacja głównej pętli odpytywania oraz funkcji wywoływanej na odebranej odpowiedzi
— modbus_rtu.c i modbus_rtu.h - implementacja funkcji związanych z obsługą protokołu modbus RTU
— crc.c[1963] i crc.h[1964] - implementacja obliczania CRC pochodzi z CRC Implementation Code in C by Michael Barr[1965] (kod źródłowy[1966] dystrybuowany na zasadach public domain) i została jedynie okrojona z niewykorzystywanych w tym projekcie funkcji.
— zabbix.c i zabbix.c - implementacja sendera zabbixowego

— error_reporting.h - plik nagłówkowy definiujący makra odpowiedzialne za wyświetlanie bądź zapisywanie do logu systemowego informacji zgłaszanych przez program
— Makefile[1967]

programy pomocnicze:

— install_and_run.sh[1968] - skrypt używany do zainstalowania i uruchomienia systemu na platformie NPE (podpinany do skryptu startowego systemu)
— modbus_reader.sql - konfiguracja SQL-owa programu określająca jakie rejestry, jakich urządzeń i jak często mają być czytane. Przykładowa konfiguracja obsługuje odczyt prawie wszystkich rejestrów analizatorów PM710 z rozróżnieniem czasowym (rejestry związane z aktualną wartością prądu czytane co 15 sekund, a najbardziej wolno zmienne co 5 minut).
— make.sh, body.template.xml, header.xml, footer.xml - system generujący szablon do zabbixa w oparciu o plik konfiguracji sql-owej systemu
— analizatory.php - skrypt php wyciągający z bazy zabbixowej ostatnie wartości dla wskazanego analizatora PM710 i prezentujący je na WWW
— monitor_modbus.sh - skrypt monitorujący wartość graniczne wybranych parametrów w oparciu o lokalna bazę sqlite
— modbus_listener.c - pomocnicze narzędzie umożliwiające podsłuch linii modbusowej
— modbus_read_write.c - prosty program odczytujący podany rejestr modbusowy, bądź zapisujący do niego podaną wartość

Pobierz[1969]
Koncepcje dalszego rozwoju
— zastosowanie std:map z C++ do przechowywania ostatnich wartości + dostęp do nich z cmdline + analiza progów alarmowych - szczegóły w modbus_reader.c
Zobacz też: Specyfikacja Modbus Application Protocol[1970]

MODBUS_READER.C

```
#include <inttypes.h>
#include <stdio.h>

#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/time.h>
#include <math.h>

#include <netinet/in.h>
#include <arpa/inet.h>

#include <sqlite3.h>

#include "error_reporting.h"
#include "modbus_rtu.h"
#include "zabbix.h"


#define DATABESE_FILE "/tmp/modbus.db"

#define MODBUS_RETRY 3
#define MODBUS_READ_BUF_SIZE 255
uint8_t modbus_read_buf[MODBUS_READ_BUF_SIZE];

#define ZABBIX_SERWER_IP "192.168.119.13"
#define ZABBIX_SERWER_PORT 10051
#define ZABBIX_MESSAGE_BUF_SIZE 8192
char zabbix_message_buf[ZABBIX_MESSAGE_BUF_SIZE];
```

```
#define LOCALDB_CONDITION ( reg==1042 || reg==1044 || reg==1046 || reg==1060 || reg==1062 || ⤸
reg==1064 )

void parse_modbus_value(uint8_t addr, uint16_t reg, float val, const char *dev_name, int ⤸
reg_list,
                        sqlite3 *db, zabbix_msg_t *msg, struct timeval *tv) {
 char buf[100];

if LOCALDB_CONDITION {
  //snprintf(buf, 100, "INSERT INTO data VALUES (%d, %d, %f, %d);", addr, reg, val, (int)tv->⤸
  tv_sec);
  snprintf(buf, 100, "UPDATE data SET val=%f, time_s=%d WHERE address=%d AND register=%d;",
           val, (int)tv->tv_sec, addr, reg);
  char *ErrMsg = NULL;
  if ( sqlite3_exec(db, buf, NULL, NULL, &ErrMsg) != SQLITE_OK ) {
   LOG_PRINT_WARN("Adding data to database - SQL error: %s\n", ErrMsg);
   sqlite3_free(ErrMsg);
  }
 }

/* zamiast robienia UPDATE w sql można rozważyć trzymanie tych danych w pamięci programu ⤸
 z uzyciem std::map z C++:
  *    - mamy tablicę indeksowaną adresami urządzeń
  *    - tablica ta zawiera wskaźniki do struktur opisujących dane urządzenie
  *    - struktura taka zawiera mapę rejestr-wartość
  * program okresowo (osobny wątek)m sprawdza czy ostatnie wartości nie przekraczają ⤸
  progów alarmowych
  * progi alarmowe przechowywane również w mapach znajdujących się w tych strukturach
  *
  * możemy także rozważyć trzymanie w tych strukturach danych dotyczących sposobu ⤸
  czytania urządzenia
  */

snprintf(buf, 100, "pm710.%d", reg);
 zabbix_message_add_item_f(msg, dev_name, buf, val);
}

int main(int argc, char **argv) {
 // open syslog
 openlog("modbus_reader", LOG_PID, LOG_DAEMON);

// initialize RS485
 int serial = init_tty("/dev/ttyS3", B19200);
 if ( serial < 0 ) {
  LOG_PRINT_CRIT("Open TTY ERROR\n");
  exit(-1);
 }

// open database
 sqlite3 *db;
 if ( sqlite3_open(DATABESE_FILE, &db) ) {
  LOG_PRINT_CRIT("Can't open database: %s", sqlite3_errmsg(db));
  sqlite3_close(db);
  exit(-1);
 }

// prepare zabbix message structure
 zabbix_msg_t message;
 struct sockaddr_in zabbix_ser;
 zabbix_ser.sin_family=AF_INET;
```

```
 zabbix_ser.sin_port=htons(ZABBIX_SERWER_PORT);
 zabbix_ser.sin_addr.s_addr=inet_addr(ZABBIX_SERWER_IP);

message.zabbix_server = &zabbix_ser;
 message.max_size = ZABBIX_MESSAGE_BUF_SIZE;
 message.data = zabbix_message_buf;

// start loop
 int count = 0;
 struct timeval loop_time;
 loop_time.tv_usec = 500000;
 loop_time.tv_sec = 0;
 while(1) {
  // zapis czasu
  struct timeval tv_curr, tv_next, timeout;
  gettimeofday(&tv_curr, NULL);
  timeradd(&loop_time, &tv_curr, &tv_next);

  int count2 = 0, count3 = 0;

  // operacje na szynie modbusowej
  int ret;
  sqlite3_stmt *stmt;
  ret = sqlite3_prepare_v2( db,
   "SELECT d.address, d.name, r.base_register, r.offset, r.function, r.value_type, r.
   reg_list_id" \
   " FROM devices AS d" \
   " JOIN device_groups AS g ON" \
   "   ( d.dev_group = g.dev_group AND ((?1 % g.frenq) = (d.frenq_offset*g.
   device_offset_multip + g.frenq_offset)) )" \
   " JOIN reg_list AS r ON" \
   "   (g.reg_list_id = r.reg_list_id)" \
   " ORDER BY d.address,r.base_register;",
   -1, &stmt, NULL
  );
  if ( ret != SQLITE_OK ) {
   LOG_PRINT_WARN("sqlite3_prepare_v2() in %s:%d exit with code %d", __FILE__, __LINE__, ret);
  } else {
   ret = sqlite3_bind_int(stmt, 1, count++);
   if ( ret != SQLITE_OK ) {
    LOG_PRINT_WARN("sqlite3_bind_int() in %s:%d exit with code %d", __FILE__, __LINE__, ret);
   } else {
    while ( ( ret = sqlite3_step(stmt) ) == SQLITE_ROW ) {
     /// petla po czytanych z urządzeń grupach rejestrów dla danego obiegu petli głównej
     (count) -->
     int address = sqlite3_column_int(stmt, 0);
     const char *device_name = (const char*) sqlite3_column_text(stmt, 1);
     int base_register = sqlite3_column_int(stmt, 2);
     int offset = sqlite3_column_int(stmt, 3);
     int function = sqlite3_column_int(stmt, 4);
     int value_type = sqlite3_column_int(stmt, 5);
     int reg_list_id = sqlite3_column_int(stmt, 6);

     int i = 0;
     count2++;
     do {
      count3++;
      ret = modbus_write_read(serial, address, function, base_register - 1,
                              offset, modbus_read_buf, MODBUS_READ_BUF_SIZE, 0);
      gettimeofday(&tv_curr, NULL);
```

```
      } while (ret < 0 && i++ < MODBUS_RETRY);

      if (ret < 0) {
       LOG_PRINT_ERROR("modbus_write_read() in %s:%d for addr=%d, base_reg=%d exit with code ↩
       %d",
              __FILE__, __LINE__, address, base_register, ret);
      } else {
       zabbix_message_reinit(&message);
       switch(value_type) {
        case 1: // float
         for (offset=0; offset < ret; offset++) {
          uint8_t raw_val[4];
          if(offset%4 == 0) {
           raw_val[3] = modbus_read_buf[offset];
           raw_val[2] = modbus_read_buf[offset+1];
           raw_val[1] = modbus_read_buf[offset+2];
           raw_val[0] = modbus_read_buf[offset+3];
           if ( ! isnan(*(float *)raw_val) )
            parse_modbus_value(address, base_register+offset/2, *(float *)raw_val,
                               device_name, reg_list_id, db, &message, &tv_curr);
          }
         }
         break;;
        case 2: // integer
         for (offset=0; offset < ret; offset++) {
          if(offset%2 == 0) {
           parse_modbus_value(address, base_register+offset,
                              ntohs(*(uint16_t *)(modbus_read_buf+offset)),
                              device_name, reg_list_id, db, &message, &tv_curr);
          }
         }
         break;;
        default:
         LOG_PRINT_WARN("Unsupported modbus value_type");
       }
       if ( zabbix_send(&message) < 0 )
        LOG_PRINT_ERROR("ERROR in zabbix_send");
      }
      /// <-
     }
     if ( ret != SQLITE_DONE ) {
      LOG_PRINT_WARN("sqlite3_step() in %s:%d exit with code %d", __FILE__, __LINE__, ret);
     }
    }
   }
   if ( ( ret = sqlite3_finalize(stmt) ) != SQLITE_OK ) {
    LOG_PRINT_WARN("sqlite3_finalize() in %s:%d exit with code %d", __FILE__, __LINE__, ret);
   }

   // porównanie czasu z zapisanym i czekanie na select()
   gettimeofday(&tv_curr, NULL);
   timersub(&tv_next, &tv_curr, &timeout);
   if (timeout.tv_sec < 0)
    LOG_PRINT_WARN("OverRun Error (%ds %dus) in count=%d count2=%d count3=%d",
           (int)timeout.tv_sec, (int)timeout.tv_usec, count, count2, count3);
   else
    select(0, NULL, NULL, NULL, &timeout);
  }

 sqlite3_close(db);
```

```
 close(serial);
 closelog();
 return 0;
}
```

MODBUS_RTU.C

```
#include <inttypes.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <termios.h>
#include <netinet/in.h>

#include "error_reporting.h"

#include "crc.h"
inline uint16_t modbus_check_sum(uint8_t *data, uint16_t buf_pos) {
 return crcSlow((unsigned char* const)data, (int)buf_pos);
}

#define TTY_READ_TIMEOUT_1 100000
#define TTY_READ_TIMEOUT_2  40000

int init_tty(const char *tty_device, int tty_baud_flag) {
 struct termios term;
 int tty_fd;

tty_fd = open(tty_device, O_RDWR | O_NOCTTY | O_SYNC);
 if (tty_fd < 0) {
  LOG_PRINT_ERROR("Opening serial port: %m");
  return tty_fd;
 } else {
  tcflush(tty_fd, TCIFLUSH);
  //fcntl(tty_fd, F_SETFL, FASYNC);

  memset(&term, 0, sizeof(term));

  term.c_iflag = IGNBRK | IGNPAR /* bez parzystości */;
  term.c_oflag = term.c_lflag = term.c_line = 0;
  term.c_cflag = CREAD |  CLOCAL | tty_baud_flag | \
   CS8 /* 8 bitow */;
  term.c_cc[VMIN]=1;
  term.c_cc[VTIME]=0;
  tcsetattr (tty_fd, TCSAFLUSH, &term);

  tcflush(tty_fd, TCIOFLUSH);
 }

return tty_fd;
}

int modbus_write_read(int tty_fd, uint8_t addr, uint8_t func, uint16_t start_reg, uint8_t ⤸
num_reg,
                      void *data, uint16_t data_len, uint8_t verbose) {
 uint8_t buft[260];
 uint8_t* buf = buft;
```

```
 int buf_pos; // pozycja w buforze (W BAJTACH)
 uint16_t check_sum, i;
 int ret;
 struct timeval timeout;
 fd_set tty_fd_set;


/// prepare modbus frame
 buf_pos = 0;
 buf[buf_pos++] = addr;
 buf[buf_pos++] = func;
 *(uint16_t*)(buf + buf_pos) = htons(start_reg);
 buf_pos += 2;

if (func == 0x03 || func == 0x04) { /// read
  buf[buf_pos++] = 0;
  buf[buf_pos++] = num_reg;
 } else if (func == 0x10) {          /// write multiple
  uint16_t *reg_value = (uint16_t *)data;
  buf[buf_pos++] = 0;
  buf[buf_pos++] = num_reg;
  for (i=0; i<num_reg; i++) {
   *(uint16_t*)(buf + buf_pos) = htons(reg_value[i]);
   buf_pos += 2;
  }
 } else if (func == 0x06) {          /// write single
  *(uint16_t*)(buf + buf_pos) = htons( *(uint16_t *)data );
  buf_pos += 2;
 } else {
  return -10;
 }

*(uint16_t*)(buf + buf_pos) = modbus_check_sum(buf, buf_pos);
 buf_pos += 2;


/// sending modbus frame
#ifdef DEBUG
 uint8_t tmpbuf[1310];
 for (i=0; i < buf_pos; i++)
  sprintf(tmpbuf + i*5, "0x%02x ", buf[i]);
 LOG_PRINT_DEBUG("REQUEST: %s", tmpbuf);
#endif

ret = write(tty_fd, buf, buf_pos);
 if (ret != buf_pos) {
  LOG_PRINT_WARN("Error write data from device: %m");
  return -11;
 }


/// receive modbus response
 FD_ZERO(&tty_fd_set);
 FD_SET(tty_fd, &tty_fd_set);
 timeout.tv_sec = 0;
 timeout.tv_usec = TTY_READ_TIMEOUT_1;
 ret = select(tty_fd+1, &tty_fd_set, NULL, NULL, &timeout);
#ifdef DEBUG
 LOG_PRINT_DEBUG("SELECT RETURN: %d", ret);
#endif
```

```
 if (ret > 0) {
  buf_pos = read(tty_fd, buf, 260);
  if (buf_pos < 0) {
   LOG_PRINT_WARN("Error read data from device - read: %m");
   return -21;
  }
  // doczytaie reszty pakietu
  do {
   FD_ZERO(&tty_fd_set);
   FD_SET(tty_fd, &tty_fd_set);
   timeout.tv_sec = 0;
   timeout.tv_usec = TTY_READ_TIMEOUT_2;
   ret = select(tty_fd+1, &tty_fd_set, NULL, NULL, &timeout);
   if (ret>0) {
    buf_pos += read(tty_fd, buf+buf_pos, 260-buf_pos);
    if (buf_pos < 0) {
     LOG_PRINT_WARN("Error read data from device - read (2): %m");
     return -22;
    }
   }
  } while (ret>0);
 } else if (ret == 0) {
  LOG_PRINT_WARN("Timeout on read from device: %d, start_reg: %d", addr, start_reg);
  return -20;
 } else {
  LOG_PRINT_WARN("Error read data from device - select: %m");
  return -23;
 }

while (buf[0] == 0) {
  buf++;
  buf_pos--;
 }

/// checking modbus response
 if (buf_pos < 3) {
  LOG_PRINT_WARN("Too short response from device: %d, start_reg: %d", addr, start_reg);
  return -24;
 }
 check_sum = modbus_check_sum(buf, buf_pos-2);

#ifdef DEBUG
 for (i=0; i < buf_pos; i++)
  sprintf(tmpbuf + i*5, "0x%02x ", buf[i]);
 LOG_PRINT_DEBUG("RESPONSE: %s", tmpbuf);
 LOG_PRINT_DEBUG("RESPONSE CHECK SUM: 0x%04x\n", check_sum);
#endif

if ( *(uint16_t*)(buf+buf_pos-2) != check_sum) {
  LOG_PRINT_WARN("Error read data (reg=%d) from device %d - check_sum", start_reg, addr);
  return -30;
 }

if ( buf[1] != func ) {
  LOG_PRINT_WARN("Error read data (reg=%d) from device %d - modbus error code: %d", start_reg,↩
  addr, buf[2]);
  return -31;
 }

if (func == 0x03 || func == 0x04) {
```

```
  for (i=0; i < buf[2]; i++) {
   if (i == data_len) {
    LOG_PRINT_WARN("Output Buffer Overflow");
    return -32;
   } else {
    *(uint8_t *)(data+i) = buf[i+3];
   }
  }
  return buf[2];
 }

return 0;
}
```

MODBUS_RTU.H

```
#ifndef MODBUS_RTU_H
#define MODBUS_RTU_H

#include <inttypes.h>
#include <termios.h>

int init_tty(const char *tty_device, int tty_baud_flag);

int modbus_write_read(int tty_fd, uint8_t addr, uint8_t func, uint16_t start_reg, uint8_t ⤸
num_reg, void *data, uint16_t data_len, uint8_t verbose);

#endif
```

MODBUS_READER.SQL

```
CREATE TABLE devices (address INT, dev_group INT, frenq_offset INT, name TEXT);
CREATE TABLE device_groups (dev_group INT, reg_list_id INT, frenq INT, frenq_offset INT, ⤸
device_offset_multip INT, name TEXT, PRIMARY KEY (dev_group, reg_list_id));

CREATE TABLE reg_list (reg_list_id INT PRIMARY KEY, base_register INT, offset INT, function ⤸
INT, value_type INT, desc TEXT);
CREATE TABLE reg_desc (reg_list_id INT, register INT, desc TEXT, units TEXT, PRIMARY KEY (⤸
reg_list_id, register));

CREATE TABLE data (address INT, register INT, val REAL, time_s INT);



INSERT INTO reg_list VALUES (1, 1000, 34, 3, 1, "PM710 Total - Average and Demand");
INSERT INTO reg_desc VALUES (1, 1000, "Real Energy Consumption", "kWh");
INSERT INTO reg_desc VALUES (1, 1002, "Apparent Energy Consumption", "kVAh");
INSERT INTO reg_desc VALUES (1, 1004, "Reactive Energy Consumption", "kVARh");
INSERT INTO reg_desc VALUES (1, 1006, "Real Power - Total", "kW");
INSERT INTO reg_desc VALUES (1, 1008, "Apparent Power - Total", "kVA");
INSERT INTO reg_desc VALUES (1, 1010, "Reactive Power - Total", "kVAR");
INSERT INTO reg_desc VALUES (1, 1012, "Power Factor - Total", "");
INSERT INTO reg_desc VALUES (1, 1014, "Voltage L-L - 3P Average", "V");
INSERT INTO reg_desc VALUES (1, 1016, "Voltage L-N - 3P Average", "V");
INSERT INTO reg_desc VALUES (1, 1018, "Current - 3P Average", "A");
```

```sql
INSERT INTO reg_desc VALUES (1, 1020, "Frequency", "Hz");
INSERT INTO reg_desc VALUES (1, 1022, "Real Power - Total Demand Present", "kW");
INSERT INTO reg_desc VALUES (1, 1024, "Apparent Power - Total Demand Present", "kVA");
INSERT INTO reg_desc VALUES (1, 1026, "Reactive Power - Total Demand Present", "kVAR");
INSERT INTO reg_desc VALUES (1, 1028, "Real Power - Total Demand Peak", "kW");
INSERT INTO reg_desc VALUES (1, 1030, "Apparent Power - Total Demand Peak", "kVA");
INSERT INTO reg_desc VALUES (1, 1032, "Reactive Power - Total Demand Peak", "kVAR");

INSERT INTO reg_list VALUES (10, 1034, 8, 3, 1, "PM710 Current");
INSERT INTO reg_desc VALUES (10, 1034, "Current A", "A");
INSERT INTO reg_desc VALUES (10, 1036, "Current B", "A");
INSERT INTO reg_desc VALUES (10, 1038, "Current C", "A");
INSERT INTO reg_desc VALUES (10, 1040, "Current N", "A");

INSERT INTO reg_list VALUES (2, 1042, 12, 3, 1, "PM710 Demand Current");
INSERT INTO reg_desc VALUES (2, 1042, "Current A - Demand Present", "A");
INSERT INTO reg_desc VALUES (2, 1044, "Current B - Demand Present", "A");
INSERT INTO reg_desc VALUES (2, 1046, "Current C - Demand Present", "A");
INSERT INTO reg_desc VALUES (2, 1048, "Current A - Demand Peak", "A");
INSERT INTO reg_desc VALUES (2, 1050, "Current B - Demand Peak", "A");
INSERT INTO reg_desc VALUES (2, 1052, "Current C - Demand Peak", "A");

INSERT INTO reg_list VALUES (9, 1054, 12, 3, 1, "PM710 Voltage");
INSERT INTO reg_desc VALUES (9, 1054, "Voltage A-B", "V");
INSERT INTO reg_desc VALUES (9, 1056, "Voltage B-C", "V");
INSERT INTO reg_desc VALUES (9, 1058, "Voltage C-A", "V");
INSERT INTO reg_desc VALUES (9, 1060, "Voltage A-N", "V");
INSERT INTO reg_desc VALUES (9, 1062, "Voltage B-N", "V");
INSERT INTO reg_desc VALUES (9, 1064, "Voltage C-N", "V");

INSERT INTO reg_list VALUES (3, 1066, 18, 3, 1, "PM710 Power");
INSERT INTO reg_desc VALUES (3, 1066, "Real Power A", "kW");
INSERT INTO reg_desc VALUES (3, 1068, "Real Power B", "kW");
INSERT INTO reg_desc VALUES (3, 1070, "Real Power C", "kW");
INSERT INTO reg_desc VALUES (3, 1072, "Apparent Power A", "kVA");
INSERT INTO reg_desc VALUES (3, 1074, "Apparent Power B", "kVA");
INSERT INTO reg_desc VALUES (3, 1076, "Apparent Power C", "kVA");
INSERT INTO reg_desc VALUES (3, 1078, "Reactive Power A", "kVAR");
INSERT INTO reg_desc VALUES (3, 1080, "Reactive Power B", "kVAR");
INSERT INTO reg_desc VALUES (3, 1082, "Reactive Power C", "kVAR");

INSERT INTO reg_list VALUES (4, 1084, 20, 3, 1, "PM710 THD");
INSERT INTO reg_desc VALUES (4, 1084, "THD @ Current A", "%");
INSERT INTO reg_desc VALUES (4, 1086, "THD @ Current B", "%");
INSERT INTO reg_desc VALUES (4, 1088, "THD @ Current C", "%");
INSERT INTO reg_desc VALUES (4, 1092, "THD @ Voltage A-N", "%");
INSERT INTO reg_desc VALUES (4, 1094, "THD @ Voltage B-N", "%");
INSERT INTO reg_desc VALUES (4, 1096, "THD @ Voltage C-N", "%");
INSERT INTO reg_desc VALUES (4, 1098, "THD @ Voltage A-B", "%");
INSERT INTO reg_desc VALUES (4, 1100, "THD @ Voltage B-C", "%");
INSERT INTO reg_desc VALUES (4, 1102, "THD @ Voltage C-A", "%");

INSERT INTO reg_list VALUES (5, 1104, 30, 3, 1, "PM710 minimum");
INSERT INTO reg_desc VALUES (5, 1104, "Minimum Real Power - Total", "kW");
INSERT INTO reg_desc VALUES (5, 1106, "Minimum Apparent Power - Total", "kVA");
INSERT INTO reg_desc VALUES (5, 1108, "Minimum Reactive Power - Total", "kVAR");
INSERT INTO reg_desc VALUES (5, 1110, "Minimum Power Factor - Total", "");
INSERT INTO reg_desc VALUES (5, 1112, "Minimum Frequency", "Hz");
INSERT INTO reg_desc VALUES (5, 1114, "Minimum Current A", "A");
INSERT INTO reg_desc VALUES (5, 1116, "Minimum Current B", "A");
```

```
INSERT INTO reg_desc VALUES (5, 1118, "Minimum Current C", "A");
INSERT INTO reg_desc VALUES (5, 1120, "Minimum Current N", "A");
INSERT INTO reg_desc VALUES (5, 1122, "Minimum Voltage A-N", "V");
INSERT INTO reg_desc VALUES (5, 1124, "Minimum Voltage B-N", "V");
INSERT INTO reg_desc VALUES (5, 1126, "Minimum Voltage C-N", "V");
INSERT INTO reg_desc VALUES (5, 1128, "Minimum Voltage A-B", "V");
INSERT INTO reg_desc VALUES (5, 1130, "Minimum Voltage B-C", "V");
INSERT INTO reg_desc VALUES (5, 1132, "Minimum Voltage C-A", "V");

INSERT INTO reg_list VALUES (6, 1134, 20, 3, 1, "PM710 minimum THD");
INSERT INTO reg_desc VALUES (6, 1134, "Minimum THD @ Current A", "%");
INSERT INTO reg_desc VALUES (6, 1136, "Minimum THD @ Current B", "%");
INSERT INTO reg_desc VALUES (6, 1138, "Minimum THD @ Current C", "%");
INSERT INTO reg_desc VALUES (6, 1142, "Minimum THD @ Voltage A-N", "%");
INSERT INTO reg_desc VALUES (6, 1144, "Minimum THD @ Voltage B-N", "%");
INSERT INTO reg_desc VALUES (6, 1146, "Minimum THD @ Voltage C-N", "%");
INSERT INTO reg_desc VALUES (6, 1148, "Minimum THD @ Voltage A-B", "%");
INSERT INTO reg_desc VALUES (6, 1150, "Minimum THD @ Voltage B-C", "%");
INSERT INTO reg_desc VALUES (6, 1152, "Minimum THD @ Voltage C-A", "%");

INSERT INTO reg_list VALUES (7, 1154, 30, 3, 1, "PM710 maximum");
INSERT INTO reg_desc VALUES (7, 1154, "Maximum Real Power - Total", "kW");
INSERT INTO reg_desc VALUES (7, 1156, "Maximum Apparent Power - Total", "kVA");
INSERT INTO reg_desc VALUES (7, 1158, "Maximum Reactive Power - Total", "kVAR");
INSERT INTO reg_desc VALUES (7, 1160, "Maximum Power Factor - Total", "");
INSERT INTO reg_desc VALUES (7, 1162, "Maximum Frequency", "Hz");
INSERT INTO reg_desc VALUES (7, 1164, "Maximum Current A", "A");
INSERT INTO reg_desc VALUES (7, 1166, "Maximum Current B", "A");
INSERT INTO reg_desc VALUES (7, 1168, "Maximum Current C", "A");
INSERT INTO reg_desc VALUES (7, 1170, "Maximum Current N", "A");
INSERT INTO reg_desc VALUES (7, 1172, "Maximum Voltage A-N", "V");
INSERT INTO reg_desc VALUES (7, 1174, "Maximum Voltage B-N", "V");
INSERT INTO reg_desc VALUES (7, 1176, "Maximum Voltage C-N", "V");
INSERT INTO reg_desc VALUES (7, 1178, "Maximum Voltage A-B", "V");
INSERT INTO reg_desc VALUES (7, 1180, "Maximum Voltage B-C", "V");
INSERT INTO reg_desc VALUES (7, 1182, "Maximum Voltage C-A", "V");

INSERT INTO reg_list VALUES (8, 1184, 20, 3, 1, "PM710 maximum THD");
INSERT INTO reg_desc VALUES (8, 1184, "Maximum THD @ Current A", "%");
INSERT INTO reg_desc VALUES (8, 1186, "Maximum THD @ Current B", "%");
INSERT INTO reg_desc VALUES (8, 1188, "Maximum THD @ Current C", "%");
INSERT INTO reg_desc VALUES (8, 1192, "Maximum THD @ Voltage A-N", "%");
INSERT INTO reg_desc VALUES (8, 1194, "Maximum THD @ Voltage B-N", "%");
INSERT INTO reg_desc VALUES (8, 1196, "Maximum THD @ Voltage C-N", "%");
INSERT INTO reg_desc VALUES (8, 1198, "Maximum THD @ Voltage A-B", "%");
INSERT INTO reg_desc VALUES (8, 1200, "Maximum THD @ Voltage B-C", "%");
INSERT INTO reg_desc VALUES (8, 1202, "Maximum THD @ Voltage C-A", "%");
```

– zakladamy:
– * obieg petli: 0.5s
– * ilosc tranzakcji modbus w obiegu petli: 3 ==> 3 sloty: A, B, C)
– * ilosc urzadzen: 30 ==> dzielnik modulo jest wielokrotnoscia 30, ofsety sa odlegle o 30

– 15s (cnt % 30) ==> slot A

```
INSERT INTO device_groups VALUES (710, 10,  30,   0, 1, "");
```

```
-- 30s (cnt % 60) ==> slot B
INSERT INTO device_groups VALUES (710,  9,  60,   0, 1, "");
INSERT INTO device_groups VALUES (710,  3,  60,  30, 1, "");

-- 60s (cnt % 120) ==> 1/2 slotu C
INSERT INTO device_groups VALUES (710,  1, 120,   0, 1, "");
INSERT INTO device_groups VALUES (710,  4, 120,  30, 1, "");

-- 5min. (low priority) ==> 1/4 slotu C
-- for (i=0;i<50;i++) if (i*30 % 120 == 60 || i*30 % 120 == 90) printf("%d\t%d\t%d\t%d\t%d\n",
i*30, i*30 % 30, i*30 % 60, i*30 % 120, i*30 % 1200)
INSERT INTO device_groups VALUES (710,  2, 600,  60, 1, "");
INSERT INTO device_groups VALUES (710,  5, 600, 180, 1, "");
INSERT INTO device_groups VALUES (710,  6, 600, 300, 1, "");
INSERT INTO device_groups VALUES (710,  7, 600, 420, 1, "");
INSERT INTO device_groups VALUES (710,  8, 600, 540, 1, "");



INSERT INTO devices VALUES ( 5, 710,  0, "RG_1A");
INSERT INTO devices VALUES (13, 710,  1, "RG_2A");

INSERT INTO devices VALUES (17, 710,  2, "RINF_1A");
INSERT INTO devices VALUES (18, 710,  3, "RINF_1B");
INSERT INTO devices VALUES (19, 710,  4, "RINF_2A");
INSERT INTO devices VALUES (20, 710,  5, "RINF_2B");
INSERT INTO devices VALUES (21, 710,  6, "RINF_3A");
INSERT INTO devices VALUES (22, 710,  7, "RINF_3B");

INSERT INTO devices VALUES (14, 710,  8, "RG_A14");

INSERT INTO devices VALUES ( 6, 710,  9, "RG_A1");
INSERT INTO devices VALUES ( 7, 710, 10, "RG_A2");
INSERT INTO devices VALUES ( 8, 710, 11, "RG_A3");
INSERT INTO devices VALUES ( 9, 710, 12, "RG_A4");
INSERT INTO devices VALUES (10, 710, 13, "RG_A5");
INSERT INTO devices VALUES (11, 710, 14, "RG_A6");

INSERT INTO devices VALUES (15, 710, 15, "RG_A15");

INSERT INTO devices VALUES (23, 710, 16, "RINF_4A");
INSERT INTO devices VALUES (24, 710, 17, "RINF_4B");
INSERT INTO devices VALUES (25, 710, 18, "RINF_5A");
INSERT INTO devices VALUES (26, 710, 19, "RINF_5B");
INSERT INTO devices VALUES (27, 710, 20, "RINF_6A");
INSERT INTO devices VALUES (28, 710, 21, "RINF_6B");

--INSERT INTO devices VALUES (12, 710, 22, "RG_A7");
INSERT INTO devices VALUES (13, 710, 25, "RG_2A");
```

MONITOR_MODBUS.SH

```
#!/bin/sh

# rozdzielana przecinkami lista odbiorców mailowych
MAIL_TO="alarmy@example.org"
```

```
# adres nadawcy
MAIL_FROM="alarmy@example.org"

# serwer pocztowy
MAIL_VIA="192.168.119.13"

# send_alert priority message_short message_full
send_alert() {
 echo "$2"

# wysyłanie maila
 echo "$3" | email -n "monitor_di.sh" -f $MAIL_FROM -r $MAIL_VIA -s "$2" $MAIL_TO

# wysyłanie SMSa poprzez centralny skrypt z systemu smtp2sms
 echo -e "Subject: Alarm z BMS\n\nSite number: NPE Priority $1\nAlarm block description: $2" ⤸
|
  /opt/smtp2sms/mail2sms.sh
}

# liczba monitorowanych pozycji
NUM=7

# ustawienie wstepne stanow na OK
for i in `seq 1 $NUM`; do
 eval STANY_$i=OK
eval INFO_$i=\"RINF$i - za duzy prad\"
 eval PRIO_$i=50
 eval MAX_$i=75
 eval MIN_$i=0
done

# prady na RINF
ADRESY_1="17, 18"
ADRESY_2="19, 20"
ADRESY_3="21, 22"
ADRESY_4="23, 24"
ADRESY_5="25, 26"
ADRESY_6="27, 28"

# prad na A14
ADRESY_7="14"
INFO_7="A14 - za duzy prad"
MAX_7=99


while true; do
 for i in `seq 1 $NUM`; do
  tmp="ADRESY_$i"
  tmp=$(C='eval "echo \$$tmp"'; eval $C)
  val=`sqlite3 /tmp/modbus.db 'SELECT MAX(val) FROM data WHERE address in ('"$tmp"') AND ⤸
  register in (1042,1044,1046)' |
   cut -f1 -d.`

  tmp="STANY_$i"
  tmp=$(C='eval "echo \$$tmp"'; eval $C)
  tmp2="MAX_$i"
  tmp2=$(C='eval "echo \$$tmp2"'; eval $C)
  tmp3="MIN_$i"
  tmp3=$(C='eval "echo \$$tmp3"'; eval $C)
```

```
  if [ "$val" -gt "$tmp2" -o "$val" -lt "$tmp3" ]; then
   if [ "$tmp" != "ALARM" ]; then
    eval STANY_$i=ALARM
    tmp="INFO_$i"
    tmp=$(C='eval "echo \$$tmp"'; eval $C)
    tmp2="PRIO_$i"
    tmp2=$(C='eval "echo \$$tmp2"'; eval $C)
    send_alert $tmp2 "$tmp, Active"
   fi
  else
   if [ "$tmp" != "OK" ]; then
    eval STANY_$i=OK
    tmp="INFO_$i"
    tmp=$(C='eval "echo \$$tmp"'; eval $C)
    tmp2="PRIO_$i"
    tmp2=$(C='eval "echo \$$tmp2"'; eval $C)
    send_alert $tmp2 "$tmp, Not Active"
   fi
  fi
 done;

sleep 300
done 2>&1 | logger -p daemon.info -t monitor_modbus.sh
```

INSTALL_AND_RUN.SH

```
#!/bin/sh

exec 1>/var/log/smtp2sms.log
exec 2>&1

BASEDIR='/mnt/nand'

ln -s $BASEDIR/modbus /opt/modbus

sqlite3 /tmp/modbus.db < /opt/modbus/modbus_reader.sql
#for addr in ‘seq 5 28‘; do for reg in ‘seq 1002 2 1202‘; do
for addr in 14 `seq 17 28`; do for reg in `seq 1042 2 1046`; do
 echo "INSERT INTO data VALUES ($addr, $reg, NULL, 0);";
done; done | sqlite3 /tmp/modbus.db

/opt/modbus/modbus_reader &
/opt/modbus/monitor_modbus.sh 2>&1 &
```

MODBUS_READ_WRITE.C

```
#include <inttypes.h>
#include <stdio.h>

#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/time.h>
#include <math.h>
#include <netinet/in.h>
```

```
#include <signal.h>

#include "modbus_rtu.h"
#include "error_reporting.h"

#define MODBUS_READ_BUF_SIZE 255
uint8_t modbus_read_buf[MODBUS_READ_BUF_SIZE];

int main(int argc, char **argv) {
 if (argc < 5) {
  fprintf(stderr, "USAGE: %s tty_device_path 2400|4800|9600|19200|38400 modbus_dev_address ⤸
  modbus_register [value]\n", argv[0]);
  exit(1);
 }
 int baud;
 switch (strtoul(argv[2], NULL, 0)) {
  case 2400:  baud = B2400; break;
  case 4800:  baud = B4800; break;
  case 9600:  baud = B9600; break;
  case 19200: baud = B19200; break;
  case 38400: baud = B38400; break;
  default:
   fprintf(stderr, "Unsupported baud value\n");
   exit(2);
 }
 int addr=strtoul(argv[3], 0, 0);
 int reg=strtoul(argv[4], 0, 0);

int fun;
 if (argc > 5) {
  int val = htons( strtoul(argv[5], 0, 0) );
  modbus_read_buf[0] = val >> 8;
  modbus_read_buf[1] = val & 0xff;
  fun = 6;
 } else {
  fun = 3;
 }

// initialize RS485
 int serial = init_tty(argv[1], baud);
 if ( serial < 0 ) {
  LOG_PRINT_CRIT("Open TTY ERROR\n");
  exit(3);
 }

// read / write data from modbus
 int ret = modbus_write_read(serial,            // SERIAL: port_descriptor
                             addr, fun, reg, 1, // MODBUS: addres, function, start_reg, ⤸
                             number_of_regs
                             modbus_read_buf, MODBUS_READ_BUF_SIZE, 0);

// print output
 if (ret < 0) {
  LOG_PRINT_ERROR("modbus_write_read() in %s:%d for addr=%d, base_reg=%d exit with code %d",
    __FILE__, __LINE__, addr, reg, ret);
  exit(4);
 } else if (fun == 3) {
  printf("OK for addr=%d, base_reg=%d: 0x%X\n", addr, reg, ntohs(*(uint16_t *)(⤸
  modbus_read_buf)));
```

```
  exit(0);
 }
}
```

MODBUS_LISTENER.C

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <inttypes.h>
#include <sys/time.h>

#include "error_reporting.h"
#include "modbus_rtu.config.h"
#include "modbus_rtu.h"

int modbus_listen(int tty_fd, int cnt_max, int speed) {
 struct timeval timeout;
 fd_set tty_fd_set;
 int cnt=0, ret;
 uint8_t ctrl=0, buf;

while (cnt_max == 0 || cnt < cnt_max) {
  FD_ZERO(&tty_fd_set);
  FD_SET(tty_fd, &tty_fd_set);

  timeout.tv_sec = 0;
  timeout.tv_usec = (1000000 * 8 * 2.5 / speed);

  ret = select(tty_fd+1, &tty_fd_set, NULL, NULL, &timeout);
  if (ret > 0) {
   ret = read(tty_fd, &buf, 1);
   if (ret < 1) {
    perror("Error read data from device - read");
    return -2;
   }
   printf("0x%02x ", buf);
   cnt++;
   ctrl = 0;
  } else if (ret == 0) {
   if (ctrl == 0) {
    ctrl=1;
    printf("\n");
   }
  } else {
   perror("Error read data from device - select");
   return -3;
  }
 }
 printf("\n");
 return 0;
}

int main(int argc, char **argv) {
 // initialize RS485
 int serial = init_tty("/dev/ttyS3", B19200);
 if ( serial < 0 ) {
  LOG_PRINT_CRIT("Open TTY ERROR");
```

```
  exit(-1);
 }

puts("LISTENING MODE");
 modbus_listen(serial, 0, 19200);
 close(serial);
 return 0;
}
```

ANALIZATORY.PHP

```
<?php
$HOST = strtr($_GET["id"], ";'\"$%", "_____");
?>
<html>
<head>
 <title>Odczyty z analizatora <?php echo $HOST; ?></title>
 <style type="text/css">
 /* <![CDATA[ */
body { background-color: #F5F5F5; }
 table { background-color: #808080; margin-top:10pt; }
 td { background-color: #E0E0E0; text-align:right; }
 th { background-color: white; font-weight:bold; text-align:center; }
 /* ]]> */
</style>
</head>
<body>
<h1 style="text-align:center">Odczyty z analizatora <?php echo $HOST; ?></h1>
<?php
$db_host = "localhost";
 $db_database="zabbix";
 $db_user="zabbix";
 $db_pass="rmc490adKz0H";

mysql_connect($db_host, $db_user, $db_pass);
 mysql_select_db($db_database);

$max_time = 0;
 $min_time = 0xffffffff;
 $cur_time=time();

$values = array();
 $item_cnt = 0;
 $graph = 0;

// dane
 $sql = "SELECT v.clock, i.itemid, i.key_, v.value FROM hosts h JOIN items i JOIN history v ⤸
 ON (i.hostid=h.hostid AND v.itemid = i.itemid AND v.clock = i.lastclock AND h.host='" . ⤸
 $HOST . "');";
 $query = mysql_db_query($db_database, $sql) or die();
 while ($item = mysql_fetch_array($query)) {
  $item_cnt++;
  if ($item["clock"] > $max_time)
   $max_time = $item["clock"];
  if ($item["clock"] < $min_time)
   $min_time = $item["clock"];

  $values[ $item["key_"] ] = $item["value"];
```

```
  if ($item["key_"] == "pm710.1006")
   $graph = $item["itemid"];
 }

// identyfikator wykresu
 $sql = "SELECT DISTINCT g.graphid FROM graphs_items gi JOIN graphs g ON (gi.graphid=g.⤸
 graphid) WHERE g.name='PM710 - moce calkowite' AND itemid=$graph;";
 $query = mysql_db_query($db_database, $sql) or die();
 while ($item = mysql_fetch_array($query)) {
  $graph = $item["graphid"];
 }

if ($item_cnt < 99) {
  echo ' <p style="text-align:center; color:red; font-weight:bold">W bazie znajduej się zbyt ⤸
  mała ilość wartości - niektóre z wartości wyświetlanych jako 0.000 mogą być ⤸
  związane z brakiem odczytu danej wartości</p>' . "\n";
 }
 if ($min_time + 60*6 < $cur_time || $max_time + 30 < $cur_time) {
  echo ' <p style="text-align:center; color:red; font-weight:bold">Niektóre z odczytanych ⤸
  danych moga być nieaktualne !!!</p>' . "\n";
 }
?>
 <p style="text-align:center"><a href="http://baszta.icm.edu.pl/zabbix/charts.php?graphid=<⤸
 ?php echo $graph; ?>">
  <img src="http://baszta.icm.edu.pl/zabbix/chart2.php?graphid=<?php echo $graph; ?>&amp;⤸
  width=550&amp;height=160&amp;period=43200" alt="moce calkowite" />
 </a></p>

<center><table width="80%">
  <tr><th width="33%"></th><th width="22%">calkowitej [kWh]</th><th width="22%">pozornej [⤸
  kVAh]</th><th width="22%">biernej [kVARh]</th></tr>
  <?php printf("<tr><th>Zużycie energii</th><td>%.3f</td><td>%.3f</td><td>%.3f</td></tr>\n",
⤸
  $values["pm710.1000"]/1000, $values["pm710.1002"]/1000, $values["pm710.1004"]/1000); ?>
 </table></center>

<center><table width="80%">
  <tr><th width="33%"></th><th width="22%">minimalne</th><th width="22%">aktualne</th><th ⤸
  width="22%">maksymalne</th></tr>
<?php
printf("\t\t<tr><th>Moc rzeczywista [kW]         </th><td>%.3f</td><td>%.3f</td><td>%.3f</td>⤸
 </tr>\n", $values["pm710.1104"]/1000, $values["pm710.1006"]/1000, $values["pm710.1154"]/1000)⤸
 ;
 printf("\t\t<tr><th>Moc pozorna [kVA]            </th><td>%.3f</td><td>%.3f</td><td>%.3f</td>⤸
 </tr>\n", $values["pm710.1106"]/1000, $values["pm710.1008"]/1000, $values["pm710.1156"]/1000)⤸
 ;
 printf("\t\t<tr><th>Moc bierna [kVAR]            </th><td>%.3f</td><td>%.3f</td><td>%.3f</td>⤸
 </tr>\n", $values["pm710.1108"]/1000, $values["pm710.1010"]/1000, $values["pm710.1158"]/1000)⤸
 ;
 printf("\t\t<tr><th>Współczynnik mocy            </th><td>%.3f</td><td>%.3f</td><td>%.3f<⤸
 /td></tr>\n", $values["pm710.1110"], $values["pm710.1012"], $values["pm710.1160"]);
 printf("\t\t<tr><th>Częstotliwość [Hz]           </th><td>%.3f</td><td>%.3f</td><td>%.3f<⤸
 /td></tr>\n", $values["pm710.1112"], $values["pm710.1020"], $values["pm710.1162"]);
 printf("\t\t<tr><th>Napięcie miedzyfazowe A-B [V]</th><td>%.3f</td><td>%.3f</td><td>%.3f<⤸
 /td></tr>\n", $values["pm710.1128"], $values["pm710.1054"], $values["pm710.1178"]);
 printf("\t\t<tr><th>Napięcie miedzyfazowe B-C [V]</th><td>%.3f</td><td>%.3f</td><td>%.3f<⤸
 /td></tr>\n", $values["pm710.1130"], $values["pm710.1056"], $values["pm710.1180"]);
 printf("\t\t<tr><th>Napięcie miedzyfazowe C-A [V]</th><td>%.3f</td><td>%.3f</td><td>%.3f<⤸
 /td></tr>\n", $values["pm710.1132"], $values["pm710.1058"], $values["pm710.1182"]);
?>
```

```
 </table></center>

<center><table width="80%">
 <tr><th width="33%"></th><th width="13%">faza A</th><th width="13%">faza B</th><th ↩
width="13%">faza C</th><th width="13%">neutralny</th><th width="13%">3 fazy</th></tr>
<?php
printf("\t\t<tr><th>Moc rzeczywista [kW]        </th><td>%.3f</td><td>%.3f</td><td>%.3f</td>↩
 <td></td><td>%.3f</td></tr>\n",
                                                $values["pm710.1066"]/1000, $values["pm710.↩
                                                1068"]/1000, $values["pm710.1070"]/1000, ↩
                                                $values["pm710.1006"]/1000);
 printf("\t\t<tr><th>Moc pozorna [kVA]              </th><td>%.3f</td><td>%.3f</td><td>%.3f</td>↩
 <td></td><td>%.3f</td></tr>\n",
                                                $values["pm710.1072"]/1000, $values["pm710.↩
                                                1074"]/1000, $values["pm710.1076"]/1000, ↩
                                                $values["pm710.1008"]/1000);
 printf("\t\t<tr><th>Moc bierna [kVAR]              </th><td>%.3f</td><td>%.3f</td><td>%.3f</td>↩
 <td></td><td>%.3f</td></tr>\n",
                                                $values["pm710.1078"]/1000, $values["pm710.↩
                                                1080"]/1000, $values["pm710.1082"]/1000, ↩
                                                $values["pm710.1010"]/1000);
 printf("\t\t<tr><th>Prąd [A]                       </th><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td>%.3f</td><td>%.3f</td></tr>\n",
                                                $values["pm710.1034"], $values["pm710.1036"], ↩
                                                $values["pm710.1038"], $values["pm710.1040"], ↩
                                                $values["pm710.1018"]);
 printf("\t\t<tr><th>Napięcie [V]                   </th><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td></td><td>%.3f</td></tr>\n",
                                                $values["pm710.1060"], $values["pm710.1062"], ↩
                                                $values["pm710.1064"], $values["pm710.1016"]);

printf("\t\t<tr><th>Minimalny prąd [A]             </th><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td>%.3f</td><td></td></tr>\n",
                                                $values["pm710.1114"], $values["pm710.1116"], ↩
                                                $values["pm710.1118"], $values["pm710.1120"]);
 printf("\t\t<tr><th>Maksymalny prąd [A]            </hd><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td>%.3f</td><td></td></tr>\n",
                                                $values["pm710.1164"], $values["pm710.1166"], ↩
                                                $values["pm710.1168"], $values["pm710.1170"]);
 printf("\t\t<tr><th>Minimalne napięcie [V]         </th><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td></td><td></td></tr>\n",
                                                $values["pm710.1122"], $values["pm710.1124"], ↩
                                                $values["pm710.1126"]);
 printf("\t\t<tr><th>Maksymalne napięcie [V]        </hd><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td></td><td></td></tr>\n",
                                                $values["pm710.1172"], $values["pm710.1174"], ↩
                                                $values["pm710.1176"]);

printf("\t\t<tr><th>THD w prądzie [%%]              </th><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td></td><td></td></tr>\n",
                                                $values["pm710.1084"], $values["pm710.1086"], ↩
                                                $values["pm710.1088"]);
 printf("\t\t<tr><th>Minimalne THD w prądzie [%%]  </hd><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td></td><td></td></tr>\n",
                                                $values["pm710.1134"], $values["pm710.1136"], ↩
                                                $values["pm710.1138"]);
 printf("\t\t<tr><th>Maksymalne THD w prądzie [%%] </th><td>%.3f</td><td>%.3f</td><td>%.3f<↩
 /td><td></td><td></td></tr>\n",
                                                $values["pm710.1184"], $values["pm710.1186"], ↩
                                                $values["pm710.1188"]);
```

```
printf("\t\t<tr><th>THD w napięciu [%%]          </th><td>%.3f</td><td>%.3f</td><td>%.3f<
/td><td></td><td></td></tr>\n",
                                              $values["pm710.1092"], $values["pm710.1094"], 
                                              $values["pm710.1096"]);
printf("\t\t<tr><th>Minimalne THD w napięciu [%%] </th><td>%.3f</td><td>%.3f</td><td>%.3f<
/td><td></td><td></td></tr>\n",
                                              $values["pm710.1142"], $values["pm710.1144"], 
                                              $values["pm710.1146"]);
printf("\t\t<tr><th>Maksymalne THD w napięciu [%%]</th><td>%.3f</td><td>%.3f</td><td>%.3f<
/td><td></td><td></td></tr>\n",
                                              $values["pm710.1192"], $values["pm710.1194"], 
                                              $values["pm710.1196"]);
?>
 </table></center>

<p style="text-align:center; color:grey; font-size:small">Aktualny czas: <?php echo strftime(
"%F %T UTC", $cur_time); ?>
      Najnowsza wartość: <?php echo strftime("%F %T UTC", $max_time); ?>
      Najstarsza wartość: <?php echo strftime("%F %T UTC", $min_time); ?>
 </p>
</body>
</html>
```

MAKE.SH

```
#!/bin/bash

grep -v 'CREATE TABLE' ../modbus_reader.sql | awk '
 $3 == "reg_list" {
  gr="";
  for (i=10; i<=NF; i++)
   gr=sprintf("%s %s", gr, $i);
 }
 $3 == "reg_desc" {
  l="";
  for (i=6; i<=NF; i++)
   l=sprintf("%s %s", l, $i);
  printf ("%s,%s\n", l, gr);
 }
' | sed -e 's#, #|#g' -e 's#^ *##g' -e 's#[);"]*##g' > reg_list.txt

cat header.xml > output.xml
export IFS="|"
while read REGISTER NAME UNIT GROUP; do
 if [ "${UNIT:0:1}" = "k" ]; then
  MULTIPLIER=1
  FORMULA=1000
  UNIT=${UNIT:1}
 else
  MULTIPLIER=0
  FORMULA=1
 fi

export REGISTER
export NAME
export UNIT
export MULTIPLIER
```

```
export FORMULA
export GROUP

envsubst '${REGISTER} ${NAME} ${UNIT} ${MULTIPLIER} ${FORMULA} ${GROUP}' < item.template.xml ⤸
>> output.xml
done < reg_list.txt
cat footer.xml >> output.xml
```

ITEM.TEMPLATE.XML

```
    <item type="2" key="pm710.${REGISTER}" value_type="0">
      <description>${NAME}</description>
      <ipmi_sensor></ipmi_sensor>
      <delay>300</delay>
      <history>90</history>
      <trends>365</trends>
      <status>0</status>
      <data_type>0</data_type>
      <units>${UNIT}</units>
      <multiplier>${MULTIPLIER}</multiplier>
      <delta>0</delta>
      <formula>${FORMULA}</formula>
      <lastlogsize>0</lastlogsize>
      <logtimefmt></logtimefmt>
      <delay_flex></delay_flex>
      <authtype>0</authtype>
      <username></username>
      <password></password>
      <publickey></publickey>
      <privatekey></privatekey>
      <params></params>
      <trapper_hosts></trapper_hosts>
      <applications>
        <application>${GROUP}</application>
      </applications>
    </item>
```

FOOTER.XML

```
      </items>
      <templates/>
      <graphs/>
      <macros/>
    </host>
  </hosts>
  <dependencies/>
</zabbix_export>
```

HEADER.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<zabbix_export version="1.0" date="13.01.11" time="14.14">
  <hosts>
```

```
 <host name="analizator PM710 full">
  <proxy_hostid>0</proxy_hostid>
  <useip>1</useip>
  <dns></dns>
  <ip>0.0.0.0</ip>
  <port>10050</port>
  <status>3</status>
  <useipmi>0</useipmi>
  <groups>
   <group>@My_Templates</group>
  </groups>
  <triggers/>
  <items>
```

OneWire dla NPE (i nie tylko)

Projekt stanowi implementację systemu odczytu temperatur poprzez magistralę 1wire i termometry z serii DS18b20 dla platformy NPE[1971]. Na system składają się następujące pliki:

- 1wire_temperatura-zabbix.c - implementuje odczyt tempertury i wysłyanie jej do zabbixa
- 1wire_temperatura.c - implementuje odczyt tempertury z termometru o adresie podanym w linii poleceń
- 1wire_rom.c - odczyt adresu urządzenia 1wire
- i2c-1wire.h - funkcje do obsługi kontrolera 1wire z interfejsem i2c (DS2482)
- zabbix.c i zabbix.c - implementacja sendera zabbixowego
- error_reporting.h - plik nagłówkowy definiujący makra odpowiedzialne za wyświetlanie bądź zapisywanie do logu systemowego informacji zgłaszanych przez program

Pobierz[1972]

1wire_temperatura.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <arpa/inet.h>


#define I2C_DEVICE "/dev/i2c-1"
#define DS2482_ADDR 0x18

#define NO_USE_SYSLOG
#include "error_reporting.h"

#include "i2c-1wire.h"

int main(int argc, char **argv) {
 int i2c_fd, ret, i, len;
 uint8_t ds18b20_addr[8];

if (argc != 9) {
  printf("USAGE: %s space sepatred 8 bytyes of ds18b20 thermometer address\n", argv[0]);
  return 1;
 }
 printf("Using 1wire address: ");
 for (i=0; i<8; i++) {
  ds18b20_addr[i]=strtoul(argv[i+1], 0, 0);
```

```
  printf("0x%X ", ds18b20_addr[i]);
 }
 printf("ON 0x%X i2c device on %s\n", DS2482_ADDR, I2C_DEVICE);

/// otwarcie i konfiguracja mosta i2c - 1wire
 i2c_fd = open_i2c_device(I2C_DEVICE, DS2482_ADDR);
 if (i2c_fd < 0) {
  printf("ERROR WHILE OPENING I2C (%d)\n", i2c_fd);
  return 3;
 }
 if (ret = i2c_smbus_write_byte(i2c_fd, DS2482_RESET)) {
  printf("ERROR IN I2C COMMUNICATION (%d)\n", ret);
  return 4;
 }
 if (ret = i2c_smbus_write_byte_data(i2c_fd, DS2482_WRITE_CONFIG, 0xe1) < 0) {
  printf("ERROR IN I2C COMMUNICATION (%d)\n", ret);
  return 5;
 }


/// polecenie konwersji temperaturty dla DS18B20
 uint8_t ow_buf[12];
 if (ret= ds2482_reset(i2c_fd)) {
  printf("ERROR IN RESET (%d)\n", ret);
  return 5;
 }

#ifdef USE_CONVERT_ALL
  ow_buf[0]=0xCC;
  ow_buf[1]=0x44;
  len=1;
 #else
  ow_buf[0]=0x55;
  for(i=0; i<8; i++){
   ow_buf[i+1] = ds18b20_addr[i];
  }
  ow_buf[9]=0x44;
  len=9;
 #endif

for (i=0; i <len; i++) {
  ret = ds2482_send_and_get(i2c_fd, ow_buf[i]);
  if (ret<0) {
   printf("ERROR IN SET/GET (%d)\n", ret);
   return 6;
  }
 }

i2c_smbus_write_byte_data(i2c_fd, DS2482_WRITE_CONFIG, 0xa5); // "power byte"

ret = ds2482_send_and_get(i2c_fd, ow_buf[len]);
 if (ret<0) {
  printf("ERROR IN SET/GET (%d)\n", ret);
  return 6;
 }


sleep(1);
```

```
/// odczyt temperaturty z DS18B20
 if (ret= ds2482_reset(i2c_fd)) {
  printf("ERROR IN RESET (%d)\n", ret);
  return 7;
 }

ow_buf[0]=0x55;
 for(i=0; i<8; i++){
  ow_buf[i+1] = ds18b20_addr[i];
 }
 ow_buf[9]=0xbe;
 ow_buf[10]=0xff;
 ow_buf[11]=0xff;

for (i=0; i <12; i++) {
  ret = ds2482_send_and_get(i2c_fd, ow_buf[i]);
  if (ret<0) {
   printf("ERROR IN SET/GET (%d)\n", ret);
   return 8;
  }
  ow_buf[i] = ret;
 }


/// parsowanie temperatury
 if (
  (ow_buf[10]==0x50 && ow_buf[11]==0x05) || \
  (ow_buf[10]==0xf8 && ow_buf[11]==0x07) || \
  (ow_buf[10]==0xff && ow_buf[11]==0xff)
 ) {
  printf("ERROR in read temperature");
  return 9;
 } else {
  float val = (short)(ow_buf[11] << 8 | ow_buf[10]) / 16.0;
  printf("TEMPERATURE: %f\n", val);
  return 0;
 }
}
```

1WIRE_TEMPERATURA-ZABBIX.C

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <arpa/inet.h>
#include <sqlite3.h>

// sqlite3 /tmp/onewire.db 'CREATE TABLE ds18b20 (A INT, B INT, C INT, D INT, E INT, F INT, G ⤸
INT, H INT, host TEXT, item TEXT);'
// sqlite3 /tmp/onewire.db 'INSERT INTO ds18b20 VALUES (40, 192, 38, 115, 2, 0, 0, 138, ⤸
"test", "test");'

#include "error_reporting.h"
#include "zabbix.h"

#define I2C_DEVICE "/dev/i2c-0"
#define DS2482_ADDR 0x18
```

```
#define DATABESE_FILE "/tmp/onewire.db"

//#define USE_CONVERT_ALL 1

#define ZABBIX_SERWER_IP "192.168.119.13"
#define ZABBIX_SERWER_PORT 10051
#define ZABBIX_MESSAGE_BUF_SIZE 8192
char zabbix_message_buf[ZABBIX_MESSAGE_BUF_SIZE];


#include "i2c-1wire.h"

typedef struct sql_callbac_parm_st sql_callbac_parm_t;
struct sql_callbac_parm_st {
 zabbix_msg_t *zabbix_msg;
 int i2c_fd;
};

int temp_conv(sql_callbac_parm_t *param, int argc, char **argv, char **argv_names) {
 char ow_buf[10];
 int i;

ds2482_reset(param->i2c_fd);

ow_buf[0]=0x55;
 for(i=0; i<8; i++){
  ow_buf[i+1] = atoi(argv[i]);
 }
 ow_buf[9]=0x44;

for (i=0; i <9; i++) {
  ds2482_send_and_get(param->i2c_fd, ow_buf[i]);
 }
 i2c_smbus_write_byte_data(param->i2c_fd, DS2482_WRITE_CONFIG, 0xa5); // "power byte"
 ds2482_send_and_get(param->i2c_fd, ow_buf[9]);

return 0;
}

int temp_conv_all(sql_callbac_parm_t *param) {
 ds2482_reset(param->i2c_fd);

ds2482_send_and_get(param->i2c_fd, 0xCC);
 i2c_smbus_write_byte_data(param->i2c_fd, DS2482_WRITE_CONFIG, 0xa5); // "power byte"
 ds2482_send_and_get(param->i2c_fd, 0x44);

return 0;
}

int temp_read(sql_callbac_parm_t *param, int argc, char **argv, char **argv_names) {
 char ow_buf[12];
 int i;

ds2482_reset(param->i2c_fd);

ow_buf[0]=0x55;
 for(i=0; i<8; i++){
  ow_buf[i+1] = atoi(argv[i]);
 }
 ow_buf[9]=0xbe;
```

```
 ow_buf[10]=0xff;
 ow_buf[11]=0xff;

for (i=0; i <12; i++) {
  ow_buf[i] = ds2482_send_and_get(param->i2c_fd, ow_buf[i]);
 }

/// parsowanie temperatury
 if (
  (ow_buf[10]==0x50 && ow_buf[11]==0x05) || \
  (ow_buf[10]==0xf8 && ow_buf[11]==0x07) || \
  (ow_buf[10]==0xff && ow_buf[11]==0xff)
 ) {
  LOG_PRINT_ERROR("ERROR in read temperature for %s %s", argv[8], argv[9]);
 } else {
  float val = (short)(ow_buf[11] << 8 | ow_buf[10]) / 16.0;
#ifdef ZABBIX_SERWER_IP
  zabbix_message_add_item_f(param->zabbix_msg, argv[8], argv[9], val);
#else
  printf("%s: %s: %f\n", argv[8], argv[9], val);
#endif
 }

return 0;
}



int main(int argc, char **argv) {
 int i2c_fd, ret;
 char *ErrMsg = NULL;

openlog("onewire_reader", LOG_PID, LOG_DAEMON);

/// otwarcie i konfiguracja mosta i2c - 1wire
 i2c_fd = open_i2c_device(I2C_DEVICE, DS2482_ADDR);
 i2c_smbus_write_byte(i2c_fd, DS2482_RESET);
 i2c_smbus_write_byte_data(i2c_fd, DS2482_WRITE_CONFIG, 0xe1);

/// open database
 sqlite3 *db;
 if ( sqlite3_open(DATABESE_FILE, &db) ) {
  LOG_PRINT_CRIT("Can't open database: %s", sqlite3_errmsg(db));
  sqlite3_close(db);
  exit(-1);
 }

/// prepare zabbix message structure
 zabbix_msg_t message;
#ifdef ZABBIX_SERWER_IP
 struct sockaddr_in zabbix_ser;
 zabbix_ser.sin_family=AF_INET;
 zabbix_ser.sin_port=htons(ZABBIX_SERWER_PORT);
 zabbix_ser.sin_addr.s_addr=inet_addr(ZABBIX_SERWER_IP);

message.zabbix_server = &zabbix_ser;
 message.max_size = ZABBIX_MESSAGE_BUF_SIZE;
 message.data = zabbix_message_buf;
#endif
```

```
/// prepare struct for sqlite3_exec callback function parametrs
 sql_callbac_parm_t param;
 param.zabbix_msg = &message;
 param.i2c_fd = i2c_fd;

while(1) {
#ifdef ZABBIX_SERWER_IP
  zabbix_message_reinit(&message);
#endif

  /// konwrsja temperatury
#ifdef USE_CONVERT_ALL
  temp_conv_all((void*)&param);
#else
  ret = sqlite3_exec(db, "SELECT A, B, C, D, E, F, G, H FROM ds18b20;",
                     (int (*)(void *, int, char **, char **))temp_conv, (void*)&param, ⤸
                     &ErrMsg);
  if ( ret != SQLITE_OK ) {
   LOG_PRINT_WARN("temp_conv SQL error: %s", ErrMsg);
   sqlite3_free(ErrMsg);
  }
#endif

  sleep(1);

  /// odczyt temperatury
  ret = sqlite3_exec(db, "SELECT A, B, C, D, E, F, G, H, host, item FROM ds18b20;",
                     (int (*)(void *, int, char **, char **))temp_read, (void*)&param, ⤸
                     &ErrMsg);
  if ( ret != SQLITE_OK ) {
   LOG_PRINT_WARN("temp_conv SQL error: %s", ErrMsg);
   sqlite3_free(ErrMsg);
  }

  // wysłanie danych do zabbix'a
#ifdef ZABBIX_SERWER_IP
  if ( zabbix_send(&message) < 0 )
   LOG_PRINT_ERROR("ERROR in zabbix_send");
#endif
  sleep(30);
 }

sqlite3_close(db);
 closelog();
 return 0;
}
```

1WIRE_ROM.C

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <arpa/inet.h>


#define I2C_DEVICE "/dev/i2c-1"
#define DS2482_ADDR 0x18
```

```
#define NO_USE_SYSLOG
#include "error_reporting.h"

#include "i2c-1wire.h"

int main(int argc, char **argv) {
 int i2c_fd, ret, i;

/// otwarcie i konfiguracja mosta i2c - 1wire
 i2c_fd = open_i2c_device(I2C_DEVICE, DS2482_ADDR);
 if (i2c_fd < 0) {
  printf("ERROR WHILE OPENING I2C (%d)\n", i2c_fd);
  return 5;
 }
 if (ret = i2c_smbus_write_byte(i2c_fd, DS2482_RESET)) {
  printf("ERROR IN I2C COMMUNICATION (%d)\n", ret);
  return 5;
 }
 if (ret = i2c_smbus_write_byte_data(i2c_fd, DS2482_WRITE_CONFIG, 0xe1) < 0) {
  printf("ERROR IN I2C COMMUNICATION (%d)\n", ret);
  return 5;
 }

/// odczyt ROMu urządzenia 1wire
 uint8_t ow_buf[12];
 if (ret= ds2482_reset(i2c_fd)) {
  printf("ERROR IN RESET (%d)\n", ret);
  return 5;
 }

ow_buf[0]=0x33;
 for (i=1; i <10; i++) {
  ow_buf[i] = 0xff;
 }
 for (i=0; i <12; i++) {
  ret = ds2482_send_and_get(i2c_fd, ow_buf[i]);
  if (ret<0) {
   printf("ERROR IN SET/GET (%d)\n", ret);
   return 6;
  }
  ow_buf[i] = ret;
 }
 printf("1-WIRE ROM ADDRESS: ");
 for (i=1; i <9; i++) {
  printf("0x%X ", ow_buf[i]);
 }
 printf("\n");

return 0;
}
```

i2c-1wire.h

```
#ifndef I2C_1WIRE_H_
#define I2C_1WIRE_H_

// wymagamy biblioteki "libi2c-dev" z http://www.lm-sensors.org
```

```c
#include <linux/i2c-dev.h>

#define DS2482_RESET               0xF0
#define DS2482_SET_READ_PTR        0xE1
#define DS2482_WRITE_CONFIG        0xD2
#define DS2482_1WIRE_RESET         0xB4
#define DS2482_1WIRE_WRITE_BYTE    0xA5
#define DS2482_DATA                0xE1

int ds2482_reset(int fd) {
 int i, status;
 if (i2c_smbus_write_byte(fd, DS2482_1WIRE_RESET))
  return -1;

for (i=0; i<10; i++) {
  usleep(2);
  status = i2c_smbus_read_byte(fd);

  if (status < 0) {
   return -2;
  } else if ((status & 0x01) == 0) { // 1-wire NOT busy
   return 0;
  }
 }

return -9;
}

int ds2482_send_and_get(int fd, char wr) {
 int i, status;
 if ( i2c_smbus_write_byte_data(fd, DS2482_1WIRE_WRITE_BYTE, wr) < 0 )
  return -1;

for (i=0; i<10; i++) {
  usleep(2);
  status = i2c_smbus_read_byte(fd);

  if (status < 0) {
   return -2;
  } else if ((status & 0x01) == 0) { // 1-wire NOT busy
   if (i2c_smbus_write_byte_data(fd, DS2482_SET_READ_PTR, DS2482_DATA) < 0)
    return -3;

   return i2c_smbus_read_byte(fd);
  }
 }

return -9;
}

int open_i2c_device(const char *dev_file, int addr) {
 // open i2c bus interface
 int fd = open(dev_file, O_RDWR);
 if (fd < 0)
  return -1;

// set slave addres
 if (ioctl(fd, I2C_SLAVE, addr) < 0)
  return -2;
```

```
return fd;
}

#endif
```

MONITOR_DI.SH

```
#!/bin/sh

# skrypt służy odczytywaniu i kontrolowaniu stanu wejść cyfrowych komputerka NPE
# odczyt i ładowanie danych do zabbixa realizowane jest z wykorzystaniem dostarczonego przez ⤸
producenta
# daemona snmp wraz z stosowną biblioteką prezentującą po tym protokole m.in. stany DI


# rozdzielana przecinkami lista odbiorców mailowych
MAIL_TO="alarmy@example.org"

# adres nadawcy
MAIL_FROM="alarmy@example.org"

# serwer pocztowy
MAIL_VIA="192.168.119.13"

# send_alert priority message_short message_full
send_alert() {
 echo "$2"

# wysyłanie maila
 echo "$3" | email -n "monitor_di.sh" -f $MAIL_FROM -r $MAIL_VIA -s "$2" $MAIL_TO

# wysyłanie SMSa poprzez centralny skrypt z systemu smtp2sms
 echo -e "Subject: Alarm z BMS\n\nSite number: NPE Priority $1\nAlarm block description: $2" ⤸
|
  /opt/smtp2sms/mail2sms.sh
}


check_mask() {
 if [ $(( $alarm_diff & $1 )) -ne 0 ]; then
  if [ $(( $alarm_mask & $1 )) -ne 0 ]; then
   msg="$msg, Not Active"
  else
   msg="$msg, Active"
  fi
  return 0;
 fi
 return 1;
}

alarm_old_mask=15
while true; do
 # echo alarm_old_mask=$alarm_old_mask

npe ?PI
alarm_mask=$?
alarm_diff=$(( $alarm_mask ^ $alarm_old_mask ))
```

```
 alarm_old_mask=$alarm_mask

# echo alarm_mask=$alarm_mask alarm_diff=$alarm_diff

if [ $alarm_diff -ne 0 ]; then
  msg="SSP, blokada"
  check_mask 0x01 && send_alert 10 "$msg"
  msg="POZAR, czujka konwencjonalna -1"
  check_mask 0x02 && send_alert 10 "$msg"
  msg="POZAR, czujka konwencjonalna -2"
  check_mask 0x04 && send_alert 10 "$msg"
fi
 sleep 1
done 2>&1 | logger -p daemon.info -t monitor_di.sh
```

PS_ON_NPE.SH

```
#!/bin/bash

# skrypt służy kontrolowaniu działania procesów (związanych z prezentowanymi na strone ⤸
systemami
# odczytu modbus'a, wysyłania smsm-ów oraz monitoringu DI) na komputerze przemysłowym NPE

run_remote_cmd_via_telnet() {
 HOST="$1"
USER="$2"
PASSWORD="$3"
CMD="$4"

expect -f - << EOF
set timeout $timeout
spawn telnet $HOST
expect {
        "Unknown host" exit
        "login: "
}
send "$USER\r"
expect "assword: "
send "$PASSWORD\r"
expect "$ "
send "$CMD\r"
expect "$ "
send "exit\r"
expect "closed by foreign host.\r"
EOF
}

check_process() {
 echo -n "$1 ... "
if echo "$PROCESS_LIST" | grep "$2" > /dev/null 2> /dev/null; then
  echo "RUNNING"
else
  echo "FAILED"
fi
}

PROCESS_LIST=`run_remote_cmd_via_telnet 192.168.119.200 user user ps`
```

```
check_process "modbus_reader" "/opt/modbus/modbus_reader"
check_process "modbus monitor" "/opt/modbus/monitor_modbus.sh"
check_process "di monitor" "/opt/di/monitor_di.sh"

check_process "smtp_listener" "mail[\t ]*/opt/smtp2sms/smtp_listener"
check_process "nullmailer" "mail[\t ]*/opt/smtp2sms/nullmailer/bin/nullmailer-send"
check_process "sms_queued" "/opt/smtp2sms/sms_queued"

#check_process "sms_queue_add" "/opt/smtp2sms/sms_queue_add"
```

KONTROLA_DOSTEPU.C

```
/*
 * System kontroli dostępu w oparciu o kartę zbliżeniową i PIN
 * z wykorzystaniem kontrolera NPE, czytnika z komunikacją szeregową
 * (RS232) i protokołem EPSO firmy Roger (np. PRT12EM)
 */
#include <inttypes.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <termios.h>
#include <syslog.h>
#include <sqlite3.h>

#define DATABESE_FILE "/tmp/access.db"

/*
rm /tmp/access.db
sqlite3 /tmp/access.db 'CREATE TABLE access (card INT, pin INT, mode INT, name TEXT);'
sqlite3 /tmp/access.db 'INSERT INTO access VALUES (123456679012, 123456, 1, "t1");'
sqlite3 /tmp/access.db 'INSERT INTO access VALUES (563399023445, 3456, 1, "t2");'
sqlite3 /tmp/access.db 'INSERT INTO access VALUES (646685564478, 3456, 1, "t3");'
*/

#define NO_USE_SYSLOG
#include "error_reporting.h"

#define TTY_READ_TIMEOUT_1 2000000
#define TTY_READ_TIMEOUT_2   40000

int init_tty(const char *tty_device, int tty_baud_flag) {
 struct termios term;
 int tty_fd;

tty_fd = open(tty_device, O_RDWR | O_NOCTTY | O_SYNC);
 if (tty_fd < 0) {
  LOG_PRINT_ERROR("Opening serial port: %m");
  return tty_fd;
 } else {
  tcflush(tty_fd, TCIFLUSH);
  //fcntl(tty_fd, F_SETFL, FASYNC);
```

```
  memset(&term, 0, sizeof(term));

  term.c_iflag = IGNBRK | IGNPAR /* bez parzystości */;
  term.c_oflag = term.c_lflag = term.c_line = 0;
  term.c_cflag = CREAD |  CLOCAL | tty_baud_flag | \
   CS8 /* 8 bitow */;
  term.c_cc[VMIN]=1;
  term.c_cc[VTIME]=0;
  tcsetattr (tty_fd, TCSAFLUSH, &term);

  tcflush(tty_fd, TCIOFLUSH);
 }

return tty_fd;
}

#define SOH 0x01
#define STX 0x02
#define ETX 0x03

uint8_t epso_checksum(uint8_t *buf, uint8_t len) {
 uint8_t i, sum = 0;
 for (i=0; i<len; i++) {
  sum ^= buf[i];
 }
 return sum | 0x20;
}

int epso_write(int tty_fd, uint8_t addr, uint8_t func, uint8_t data) {
 uint8_t buf[16], pos;
 int ret;

if (func == 0xFF) {
  pos = snprintf(buf, 15, "_S%02d%02X_X_", addr, func);
  buf[pos-2] = data;
 } else {
  pos = snprintf(buf, 15, "_S%02d%02X_%d_", addr, func, data);
 }
 buf[0] = SOH;
 buf[6] = STX;
 buf[pos-1] = ETX;
 buf[pos] = epso_checksum(buf, pos);

#ifdef DEBUG_OUT
 uint8_t i;
 for (i=0; i < pos+1; i++)
  printf("  >>> %02d: 0x%02x (%c)\n", i, buf[i], buf[i]);
 #endif

ret = write(tty_fd, buf, pos+1);
 if (ret != pos+1) {
  LOG_PRINT_WARN("Error write data from device: %m");
  return -11;
 }
 return 0;
}

int epso_write_read(int tty_fd, uint8_t addr, uint8_t func, uint8_t data, char *buf_out, ↩
uint8_t buf_len) {
 uint8_t buft[260];
```

```
 uint8_t* buf = buft;
 uint8_t check_sum, i, j;
 int buf_pos, ret;
 struct timeval timeout;
 fd_set tty_fd_set;

/// sending epso request
 ret = epso_write(tty_fd, addr, func, data);

/// receive epso response
 FD_ZERO(&tty_fd_set);
 FD_SET(tty_fd, &tty_fd_set);
 timeout.tv_sec = 0;
 timeout.tv_usec = TTY_READ_TIMEOUT_1;
 ret = select(tty_fd+1, &tty_fd_set, NULL, NULL, &timeout);
 if (ret > 0) {
  buf_pos = read(tty_fd, buf, 260);
  if (buf_pos < 0) {
   LOG_PRINT_WARN("Error read data from device - read: %m");
   return -21;
  }
  // doczytaie reszty pakietu
  do {
   FD_ZERO(&tty_fd_set);
   FD_SET(tty_fd, &tty_fd_set);
   timeout.tv_sec = 0;
   timeout.tv_usec = TTY_READ_TIMEOUT_2;
   ret = select(tty_fd+1, &tty_fd_set, NULL, NULL, &timeout);
   if (ret>0) {
    buf_pos += read(tty_fd, buf+buf_pos, 260-buf_pos);
    if (buf_pos < 0) {
     LOG_PRINT_WARN("Error read data from device - read (2): %m");
     return -22;
    }
   }
  } while (ret>0);
 } else if (ret == 0) {
  LOG_PRINT_WARN("Timeout on read from device: %d, func: 0x%02x", addr, func);
  return -20;
 } else {
  LOG_PRINT_WARN("Error read data from device - select: %m");
  return -23;
 }

while (buf[0] == 0) {
  buf++;
  buf_pos--;
 }

/// checking epso response
 if (buf_pos < 3) {
  LOG_PRINT_WARN("Too short response from device: %d, func: 0x%02x", addr, func);
  return -24;
 }
 check_sum = epso_checksum(buf, buf_pos-1);

#ifdef DEBUG_IN
 for (i=0; i < buf_pos; i++)
  printf("  <<< %02d: 0x%02x (%c)\n", i, buf[i], buf[i]);
 #endif
```

```
if ( buf[buf_pos-1] != check_sum) {
  LOG_PRINT_WARN("Error read data (func=0x%02x) from device %d - check_sum (0x%x != 0x%x)", ↩
  func, addr, buf[buf_pos-1], check_sum);
  return -30;
 }

j = -40;
 for (i=7; i < buf_pos-2; i++) {
  j = i - 7;
  if (j == buf_len) {
   LOG_PRINT_WARN("Output Buffer Overflow");
   return -32;
  } else {
   buf_out[j] = buf[i];
  }
 }

return j+1;
}

#define RED    0x4
#define GREEN  0xa
#define ORANGE 0x1

#define BUF_SIZE 255
char buf[ BUF_SIZE ];

int main(int argc, char **argv) {
 // open syslog
 openlog("modbus_reader", LOG_PID, LOG_DAEMON);

// initialize RS232
 int serial = init_tty("/dev/ttyS2", B9600);
 if ( serial < 0 ) {
  LOG_PRINT_CRIT("Open TTY ERROR\n");
  exit(-1);
 }

// open database
 sqlite3 *db;
 if ( sqlite3_open(DATABESE_FILE, &db) ) {
  syslog(LOG_CRIT, "Can't open database: %s\n", sqlite3_errmsg(db));
  sqlite3_close(db);
  exit(-1);
 }

char last_card[17], last_pin[7], *bufp, *pe, isNewData;
 last_card[0] = '\0';
 last_pin[0] = '\0';

uint64_t lastCard, dbCard;
 uint32_t lastPin, dbPin;
 int len, access, waitCounter = 0;
 const char *userName = NULL;
 lastCard = 0xffffffffffffffffULL;
 lastPin  = 0xffffffff;

len = epso_write_read(serial, 0x00, 0xB1, 62, buf, BUF_SIZE);
 len = epso_write_read(serial, 0x00, 0xe8, ORANGE, buf, BUF_SIZE);
```

```
len = epso_write_read(serial, 0x00, 0xB0, 62, buf, BUF_SIZE);

while(1) {
  len = epso_write_read(serial, 0x00, 0xA5, 62, buf, BUF_SIZE);
  buf[len]='\0';

  // printf("input status: %s\n", buf+len-2);

  if (len > 4) {
   if (buf[0] == 'R') {
    bufp = buf+1;
    bufp[16] = '\0';
    // now in bufp we have HEX coded card ID
    lastCard = strtoull(bufp, NULL, 16);

    bufp = buf+18;
    isNewData = 1;
   } else {
    bufp = buf+1;
    isNewData = 0;
   }


   if (bufp[0] != ':') {
    pe = index(bufp, ':');
    *pe = '\0';
    // now in bufp we have DEC coded pin
    lastPin = strtoul(bufp, NULL, 10);

    isNewData = 1;
   }
  } else {
   isNewData = 0;
  }

  access = -2;
  if (isNewData) { // is new data
   snprintf(buf, 255, "SELECT card, pin, mode, name FROM access WHERE card = '%llu' OR pin = ⤸
   '%u'", lastCard, lastPin);
   int ret;
   sqlite3_stmt *stmt;
   ret = sqlite3_prepare_v2( db,
    buf,
    -1, &stmt, NULL
   );

   while ( ( ret = sqlite3_step(stmt) ) == SQLITE_ROW ) {
    dbCard = sqlite3_column_int64(stmt, 0);
    dbPin  = sqlite3_column_int(stmt, 1);
    int mode = sqlite3_column_int(stmt, 2);
    userName = (const char*) sqlite3_column_text(stmt, 3);

    // printf("FIND IN DB: %s (0x%llx, %d) by (0x%llx, %d) with mode %d\n", userName, dbCard, ⤸
    dbPin, lastCard, lastPin, mode);

    if (mode==1) {                                          // if check pin AND ⤸
    card
     if (lastCard == 0xffffffffffffffffULL || lastPin == 0xffffffff) { //     if don't have ⤸
     both => still waitng
      access = -1;
```

```
      waitCounter++;
     } else {                                                          //    if have both
      waitCounter = 0;
      if (dbCard==lastCard && dbPin==lastPin) {                       //     if consistent    =>⤸
      access granted
       access = 11;
       break;
      } else {                                                        //     otherwise        =>⤸
      access denaid
       access = 0;
      }
     }
    } else {                                                              // if check pin OR ⤸
    card => access granted (because find in db)
     access = 10;
     break;
    }
   }
  } else if (waitCounter > 0) { // is waiting for new data
   if (waitCounter++ > 25) { //    is waiting timeout
    access = 3;
    waitCounter = 0;
   } else {
    access = -1;
   }
  }

  if (access >=0) {
   if (access >= 10) {
    printf("ACCESS GRANTED FOR: %s (0x%llx, %d), code=%d\n", userName, lastCard, lastPin, ⤸
    access);

    system ("npe +P01");
    epso_write_read(serial, 0x00, 0xe8, GREEN, buf, BUF_SIZE);
    sleep(3);
    epso_write_read(serial, 0x00, 0xe8, ORANGE, buf, BUF_SIZE);
    system ("npe -P01");
   } else {
    printf("ACCESS DENAID card=0x%llx / pin=%d, code=%d\n", lastCard, lastPin, access);

    epso_write_read(serial, 0x00, 0xe8, RED, buf, BUF_SIZE);
    epso_write_read(serial, 0x00, 0xB1, 62, buf, BUF_SIZE);
    sleep(1);
    epso_write_read(serial, 0x00, 0xB0, 62, buf, BUF_SIZE);
    epso_write_read(serial, 0x00, 0xe8, ORANGE, buf, BUF_SIZE);
   }
   lastCard = 0xffffffffffffffffULL;
   lastPin  = 0xffffffff;
  }

  usleep(100000);
 }

return 0;
}
```

Monitoring wizyjny - CCTV, RTSP, detekcja ruchu, nagrywanie

Klasycznym podejściem do cyfrowej rejestracji nagrań z monitoringu wideo była rejestracja w oparciu o dedykowany rejestrator sprzętowy lub komputer wyposarzony w stosowną kartę wejść wideo. Rejestracji takiej mogła toważyszyć detekcja ruchu (nagrywanie tylko wtedy kiedy obraz się zmienia) oraz przesyłanie takiego obrazu przez sieć. Uciążliwością w takim podejściu była konieczność stosowania osobnego okablowania (kabel koncentryczny 75 ohmów) lub specjalych transformatorów do przesyłu obrazu z kamer po skrętce oraz moc obliczeniowa rejestratora potrzebna do kompresji wideo lub konieczność stosowania specjalistycznych kart z kompresją sprzętową.

Od czasu popularyzacji kamer IP scenariusz ten często wygląda inaczej - obraz komperowany jest sprzętowo w samej kamerze i w ten sposób przesyłany poprzez sieć komputerową. Także detekcja ruchu i nagrywanie mogą być realizowane w samej kamerze, niestety jednak kamery które potrafiłyby zapisać filmik w oparciu o detekcję (uwzględniający pre-bufor, czyli jakiś odcinek czasu przed detekcją, a także zakończyć taką rejestrację jakiś czas po ustaniu detekcji ruchu) bezpośrednio na dysku sieciowym (np. NFS) należą do rzadkości (osobiście nie trafiłem na taką). Nawet porządne kiamery dostępne na rynku (np. seria iPolis Samsunga, na której w sporej mierze bazuje ten artykuł) potrafią jedynie wysłać obrazek (pojedyńczą klatkę) na serwer FTP. Wymusza to (jeżeli chemy mieć zdalną rejestrację w postaci filmów) albo ciągłe nagrywanie strumienia RTSP, albo jego dekodowanie i detekcję ruchu z użyciem jakiegoś softu przeznaczonego do tego (np. motion, zoneminder, …), lbo próbę implementacji tego w oparciu o bufor cylkiczny (jako pre-bufor) i wykorzystanie detekcji opartej o kamerę (informacje te dostępne są z użyciem np. http).

Odbiór/nagrywanie strumienia RTSP można zrealizować na kilka sposobów np.:

- Z uzyciem live555: `openRTSP -i rtsp://admin:4321@192.168.119.49/profile2/media.smp > OUTPUT.avi`, niestety biblioteka live555 nie obsługuje obecnie IPv6
- Z uzyciem VLC: `vlc -v "rtsp://admin:4321@192.168.1.100/profile2/media.smp" --rtsp-tcp --sout "file{mux=ts,dst=OUTPUT.mp4}" :sout-keep`, niestety VLC obecnie potrafi kożystać z RTSP tylko za pośrednictwem live555, więc ta metoda również nie obsługuje IPv6
- Z użyciem libav (ffmpeg): `avconv -i rtsp://admin:4321@192.168.1.100/profile2/media.smp -vcodec copy -acodec copy -f mp4 OUTPUT.mp4` (prostrzy aviocat nie radzi sobie z RTSP, natomiast jest w stanie odbierać strumien http wysyłany przez VLC) niestety libav nie radzi sobie z niektórymi serwerami RTSP (np. z VLC)

W obu wypadkach możemy zamiast zapisywać do zwykłego pliku wysyłać odebrane dane do potoku nazwanego (patrz mkfifo) lub na standardowe wyjście (w szczególności dane odebrane avconv możemy przekazać do VLC w celu np. wyświetlenia, ale nie zawsze się to udaje). Ponaddto w obu wypadkach możemy taki strumień przesyłać ponownie (robić za proxy). Coprawda live555 zawiera w sobie proste proxy rtsp z obsługą wielu strumieni - live555ProxyServer, ale z użyciem VLC możemy robić to przy okazji i dodatkowo wspierając (na wyjściu) IPv6. Takie rozwiązanie daje nam też większe możliwości konfiguracyjne (możemy użyć np. VLM) i zapewnia lepszą stabilność od live555ProxyServer (w prowadzonych testach po jakimś czasie nie można było się do niego podłączyć (vlc zwracał "live555 demux error: no data received in 10s, aborting").

Ogólna koncepcja systemu odpowiedzialnego na rejestrację RTSP w oparciu o detekcję robioną na kamerze może wyglądać następująco:

1. Odbieramy strumień RTSP, retransmitujemy go (np. jako http lub rtsp) celem zdalnego podglądu i równoczesnie przesyłamy do narzędzi odpowiedzialnego za ngrywanie.
2. Narzędzie takie utrzymuje bufor cykliczny (tablica + przeskakujący na jej początek gdy dojdzie do końca wskaźnik - miejsce zapisu) przechowując w nim ostatnie n sekund danych.
3. Narzędzie to samo bada stan detekcji ruchu na kamerze lub dostaje taką informację z zewnatrz (np. poprzez wykrywanie H w `wget --quiet -O - 'http://admin:4321@192.168.119.49/cgi-bin/control.cgi?msubmenu=motion&action=monitor'`) i w momęcie detekcji dokonuje zapisu buforu do pliku (nazwa tworzona w oparciu o aktualny czas i ajkiś prefix).
4. Od momentu wykrycia detekcji bierzące dane dopisywane są do tego pliku aż do upłynięcia m sekund od zakończenia detekcji. Jeżli w tym czasie zajdze detekcja nie przestajemy dopisywać danych.
5. Po upłynięciu m sekund od zakończenia detekcji kończy pisanie do pliku i zaczyna pisać do bufora cyklicznego.

Do realizacji proxy dla strumieni wideo możemy także wykożystać avserver[1973]. Umożliwia on sam w sobie realizację timeshiftu opartego na jakiemś rodzaju bufora cyklicznego, dzięki czemu może uprośckić realizację opisanego powyżej nagrywania na skutek detekcji z pre-buforem.

Koncepcje dalszego rozwoju

- próba konfiguracja avserver
- walka z bugami utrudniającymi wzajemną współpracę avconv, vlc, rtsp/h264, …

— stworzenie softu implementującego bufor kołowy (wywoływanego `cvlc rtsp://admin:4321@192.168.1.100/profile2/media.smp -\-demux=h264 \-sout="#duplicate{dst=http{mux=ts,dst=:8554/kamera},dst=rtp{mux=ts,sdp=rtsp://:5554/kamera},dst=fi :sout-keep | loop_recording`)
— poszukiwania kamery która potrafiłaby to robić normalnie (zapisywać bezpośrednio na dysk sieciowy)

BOSH_DINION.SH

```
#!/bin/bash

# set of command for Bosh Dinion IP CCTV cameras

# conversion date (or unix timestamp) to camera time stamp: 946684800 is 2000-01-01 00:00:00
# USAGE: cctv_convertDate "YYYY-mm-dd HH:MM:SS"
# USAGE: cctv_convertDate "@UNIXTIME"
cctv_convertDate() {
 printf %x $(( `date -d "$@" +%s` - 946684800 ))
}

# return session id for rtsp stream
cctv_getSession() {
 rtsp_sess=`wget -O - "http://$HOST:$PORT/rcp.⤸
 xml?command=0x0ae8&type=T_DWORD&direction=READ&protocol=TCP&num=$RAND" | xmlstarlet sel -t -⤸
 v '//rcp/result/hex'`
if [ `printf %d $rtsp_sess` -ne 0 ]; then
  echo $rtsp_sess
 else
  wget -O - "http://$HOST:$PORT/rcp.⤸
  xml?command=0xFF0C&type=P_OCTET&direction=WRITE&num=1&idstring=connect&payload=0x00010010000⤸
  000000132710000000000101000000000000004040FFFF00000000" |
   xmlstarlet sel -t -v '//rcp/sessionid'
fi
}

# execute commande (read or write) on camera
# USAGE: cctv_cmd command mode args_string
cctv_cmd() {
 wget -O - "http://$HOST:$PORT/rcp.⤸
 xml?command=$1&sessionid=`cctv_getSession`&direction=$2&$3"
# get SESSION_ID before any set/get operation because session_id can be changed on reset ⤸
 video in client
}

# command write to camera
# USAGE: cctv_set command args_string
cctv_set() {
 cctv_cmd "$1" WRITE "$2"
}

# command read from camera
# USAGE: cctv_get command args_string
cctv_get() {
 cctv_cmd "$1" READ "$2"
}

# start RTSP replay session with vlc as client
# USAGE: cctv_start host port "YYYY-mm-dd HH:MM:SS"
cctv_start() {
 HOST="$1"
```

```
PORT="$2"
RAND=$RANDOM
 vlc --rtsp-tcp --rtsp-http --rtsp-http-port $PORT "rtsp://$HOST:⤸
 $PORT/rtsp_tunnel?&inst=1&rec=1?enableaudio=1&audio_mode=0&rnd=$RAND&seek=0x`cctv_convertDate⤸
 "$3"`" &
}


# seek replay to time position
# USAGE: cctv_goToDate "YYYY-mm-dd HH:MM:SS"
# vlc can stop play after this call (when go backward?)
cctv_goToDate() {
 cctv_set  0x0905 "type=P_OCTET&payload=0x`cctv_convertDate "$1"`"

}

# seek replay to current position (live stream)
cctv_goToLive() {
 cctv_set 0x0963 "type=T_DWORD&payload=1"
}

# set play speed, 100 = normal
cctv_setSpeed() {
 cctv_set  0x0902 "type=T_INT&num=1&payload=$1"
}

# read time info from camera
cctv_getDate() {
 cctv_get  0x0905 "type=P_OCTET"

}


# get raw record info from cammera
# USAGE: cctv_getRecordInfoRaw [start_time end_time]
# optional times in "YYYY-mm-dd HH:MM:SS" format
cctv_getRecordInfoRaw() {
 if [ $# -lt 2 ]; then
  cctv_get 0x0901 "type=P_OCTET&payload=0xffffffff00000000000007ff00000000"
else
  cctv_get 0x0901 "type=P_OCTET&payload=0x`cctv_convertDate "$1"``cctv_convertDate ⤸
  "$2"`000007ff00000000"
fi
}

# get formatted record info from cammera
# USAGE: cctv_getRecordInfo [start_time end_time]
# optional times in "YYYY-mm-dd HH:MM:SS" format
cctv_getRecordInfo() {
 cctv_getRecordInfoRaw "$@" | xmlstarlet sel -t -v '//rcp/result/str' |
awk -F '  ' '{print $1, $2; print $3, $4}' |
awk --non-decimal-data '
  BEGIN {
   printf("   date and time   \ttime id \tlen [s]\t  flags  \t file id\n")
  }
  $1 != "" {
   start  = ("0x"$1$2$3$4)
   len    = ("0x"$5$6$7$8) - ("0x"$1$2$3$4)
   flags  = ("0x"$9$10$11$12)
   fileid = ("0x"$13$14$15$16)
```

```
    printf("%s\t%x\t%6d\t%s\t%s\n", strftime("%Y-%m-%d %H:%M:%S", start + 946684800), start, ⤸
    len, flags, fileid)
  }
 '
}
```

SNMP-BACnet gateway

Jest to konfigurowalna bramka pomiędzy SNMP[1974] a BACnet[1975]. Bramka służy do udostępniania informacji z urządzeń korzystających z SNMP (wersja 1 i 2c) w sieci BACnetowej - stroną nawiązującą połączenia (odpytującą) jest BACnet.

Obsługa SNMP oparta jest na bibliotece libnetsnmp[1976], natomiast obsługa BACnetu oparta jest na BACnet Stack[1977] (testowane z wersją 0.5.5).

POBIERZ[1978]

W archiwum[1979] dostępna jest pełna wersja źródeł - zawierająca stos BACnet'owy oraz linkowane statycznie pliki biblioteki libnetsnmp oraz bibliotek powiązanych.

SysMon

Jest to system skryptów służących do monitorowania pracy systemów. System wspiera:

- kontrolę dostępności hostów w oparciu o echo ICMP (ping)
- kontrolę restartu hostów (w oparciu o badanie uptime)
- kontrolę temperatury na hostach (w oparciu o ipmi)
- kontrolę braku miejsca na dystkach twardych
- kontrolę stanu dysków poprzez S.MA.R.T.
- systemach z rodziny Linux i Solaris
- powiadomienia poprzez pocztę elektroniczną oraz xmpp
- automatyzację procesu instalacji i aktualizacji
- raporty diagnostyczne systemu monitorującego

Na system składają się następujące elementy

- sysmon_server.sh - skrypt wywoływany poprzez cron i odpowiedzialny za badanie stanu hostów
- sysmon_client.sh - skrypt wywoływany zdalnie poprzez ssh na monitorowanych hostach
- sysmon_server.conf - plik konfiguracyjny dla sysmon_server.sh (ustawienia wspólne dla całości systemu)
- sysmon_client.conf - plik konfiguracyjny dla sysmon_client.sh (ustawienia zalezne od danego hosta)

sysmon_server.sh

```
#!/bin/bash

### KONFIGURACJA ###
# sciezka do katalogu skryptu
system_dir="/var/lib/sysmon"

# wlaczenie reszty konfiguracji
source $system_dir/sysmon_server.conf

### ZMIENNE TRYBU PRACY ###
CMD=$1; shift 1
RUN_TIME=`date --utc +'%F %T UTC'`
RUN_FILE="$system_dir/run.$CMD.txt"


### FUNKCJA WYSYLAJACA ###
```

```
send_info() {
 # dodanie stopki wiadomosci
 msg="$2\n---\nSYSMON on `hostname -f`\n START: $RUN_TIME\n FINISH: `date --utc +'%F %T ⤸
 UTC'`"

# SMTP
 addr_list=""
for addr in $ADMIN_SMTP; do
  addr_list="$addr_list -t $addr"
done
 if [ "$addr_list" != "" ]; then
  sendEmail -f "SysMon na $HOSTNAME <$MAIL_FROM>" $addr_list $SENDEMAIL_OPTIONS -u "$1" -m ⤸
  "$msg" >/dev/null 2>/dev/null
fi

# XMPP
 for addr in $ADMIN_XMPP; do
  echo -e "$msg" | sendxmpp -f $SENDXMPP_AUTH_FILE -s "$1 (od: SysMon na $HOSTNAME)" $addr >⤸
  /dev/null 2>/dev/null
done
}

### FUNKCJA SPRAWDZAJACA DOSTEPNOSC HOSTA ###
# $1 - host address, $2 != 0 then recheck
# modify $live $dead and $recheck
check_host_live() {
 if ping -c1 $1 > /dev/null 2> /dev/null; then
  # host zyje
  if $USE_BLOCK && grep "^$1\$" $BLOCK_FILE >/dev/null; then
   sed -i"$BLOCK_BACKUP_SURFIX" -e"/^$1\$/d" $BLOCK_FILE
   live="$live\n * $1"
  fi
 else
  # host prawdopodobnie nie zyje ... zatem
  # jezeli nie ma go w pliku blokad to
  if !($USE_BLOCK && grep "^$1\$" $BLOCK_FILE >/dev/null); then
   # rozważamy ponowną próbę
   if [ $2 != 0 ] && ($recheck || $RECHECK_ANY); then
    sleep $RECHECK_TIME
    check_host_live $1 $(( $2 - 1 ))
   # a gdy wykonaliśmy stosowną ilość prób to
   else
    # uznajemy host za martwy
    dead="$dead\n * $1"
    $USE_BLOCK && echo $1 >> $BLOCK_FILE
    recheck=false
   fi
  fi
 fi
}

### FUNKCJA SPRAWDZAJACA DOSTEPNOSC URL ###
# $1 - url
check_http_services() {
 tmp_msg=`LANG=C wget -O /dev/null "$1" 2>&1`
if [ $? -eq 0 ]; then
  # jezeli używamy pliku blokad a url był w nim to usuwamy
  if $USE_BLOCK_FOR_SERVICES && grep "^$1\$" $BLOCK_FILE >/dev/null; then
   sed -i"$BLOCK_BACKUP_SURFIX" -e"/^${1////\\/}\$/d" $BLOCK_FILE
   echo " * $1 is OK\\n"
```

```
  fi
 else
  if ! ($USE_BLOCK_FOR_SERVICES && grep "^$1\$" $BLOCK_FILE >/dev/null); then
   # generujemy info
   echo "$tmp_msg" | awk '
    $3=="ERROR" {
     $1=" * '"$1"'"
     $2="report:"
     printf("%s\\n", $0)
    }
    $4~/failed/ && '$CHECK_FOR_DEAD_SERVICES'==1 {
     printf (" * %s report: %s\\n", "'$1'", $0)
    }
   '
   # gdy używamy pliku blokad dodajemy do niego ten URL
   $USE_BLOCK_FOR_SERVICES && echo $1 >> $BLOCK_FILE
  fi
 fi

}

### FUNKCJA SPRAWDZAJACA DOSTEPNOSC USŁUGI W OPARCIU O PORT ###
# $1 - adres usługi w postaci host:port
# zarówno host jak i port mogą być określane numerycznie jak i poprzez nazwę
check_port_services() {
 if [ $CHECK_FOR_DEAD_SERVICES -eq 1 ]; then
  opt="-PN"
else
  opt=""
fi
 tmp_msg=$( LANG=C nmap "${1%:*}" -p "${1#*:}" --reason $opt | grep tcp | awk '
  $2 != "open" {
   printf(" * %s : port %s (%s)\\n", "'${1}'", $2 , $4)
  }
 ' )

if [ "$tmp_msg" != "" ]; then
  if ! ($USE_BLOCK_FOR_SERVICES && grep "^$1\$" $BLOCK_FILE >/dev/null); then
   echo "$tmp_msg"
   $USE_BLOCK_FOR_SERVICES && echo $1 >> $BLOCK_FILE
  fi
 else
  if $USE_BLOCK_FOR_SERVICES && grep "^$1\$" $BLOCK_FILE >/dev/null; then
   sed -i"$BLOCK_BACKUP_SURFIX" -e"/^$1\$/d" $BLOCK_FILE
   echo " * $1 is OK\\n"
  fi
 fi
}

### FUNKCJA WYKONUJĄCA ZDALNĄ KOMĘDĘ NA HOSCIE ###
# $1 - host address, $2 - komenda
# modify $live $dead and $recheck
do_host_remote_cmd() {
 # oczekujemy że host uznawany jest za żywy
 if ! ($USE_BLOCK_FOR_SSH && grep "^$1\$" $BLOCK_FILE >/dev/null); then
  cnt=$RECHECK_RETRY
  while ! live=`RUN_REMOTE $1 $2 2> $ERROR_TMP_FILE`; do
  # { while ! error=‘ssh $1 $2 3>&2 2>&1 1>&3‘; do ... done } 2>&1
   if [ $cnt != 0 ] && ($recheck || $RECHECK_ANY); then
    sleep $RECHECK_TIME
```

```
   cnt=$(( $cnt - 1))
  else
   # host nie odpowiada na SSH
   if [ -s $ERROR_TMP_FILE ]; then
    dead="$dead\n * $1 (`cat $ERROR_TMP_FILE | tr '\n' ' '`)"
   else
    dead="$dead\n * $1"
   fi
   recheck=false
   return -1;
  fi
 done
 return 0;
fi
return -2;
}



### ZABEZPIECZENIE PRZED ROWNOLEGLA PRACA ###
if [ -e $RUN_FILE ]; then
 . $RUN_FILE
 run_timestamp=$(( `date +%s` - $run_timestamp ))
 if [ $run_timestamp -lt 1800 ]; then
  echo "SYSMON witch $CMD is running since $run_timestamp seconds" >&2
  exit;
 fi
fi
echo "run_timestamp=\"`date +%s`\"" > $RUN_FILE

### WYKONANIE AKCJI PODANEJ W ARGUMENCIE
case "$CMD" in
 # SPRAWDZANIE CZY HOSTY ZYJA
 "ping")
  live=""; dead=""
  recheck=true
  for addr in $HOSTS $HOSTS_PING_ONLY; do
   check_host_live $addr $RECHECK_RETRY
  done

  msg=""; title=""
  if [ "$dead" != "" ]; then
   # udekorowanie wiadomosci
   msg="Next $SYSTEM_NAME hosts are down:\n$dead"
   if $USE_BLOCK; then
    msg="$msg\n\nFuture warning on this host NOT BE SEND"
    msg="$msg\nuntil host don't remove from list in $HOSTNAME:$BLOCK_FILE\nor host not be up"
   fi
   title="down"
  elif [ "$live" != "" ]; then
   # udekorowanie wiadomosci
   msg="$msg\n\nNext $SYSTEM_NAME hosts are up:\n$live"
   if [ "$title" = "" ]; then title="up"; else title="$title/up"; fi
  fi

  if [ "$msg" != "" ]; then
   # wysylanie wiadomosci
   send_info "$SYSTEM_NAME: host $title alert" "$msg"
  fi
 ;;
```

```
# SPRAWDZANIE CZY DZIAŁAJĄ USŁUGI
 "services")
  msg=""
  for url in $URL_CHECK_LIST; do
   msg="$msg`check_http_services "$url"`"
  done

  for service in $SERVICES_CHECK_LIST; do
   msg="$msg`check_port_services "$service"`"
  done

  if declare -f addional_services_check > /dev/null; then
   msg="$msg`addional_services_check`"
  fi

  if [ "$msg" != "" ]; then
   # wysylanie wiadomosci
   send_info "$SYSTEM_NAME: services info/alert" "$msg"
  fi
 ;;

# SPRAWDZANIE CZY HOSTY NIE BYLY RESETOWANE
 "uptime")
  mv -f $UPTIME_FILE.new $UPTIME_FILE.old 2>/dev/null
  :> $UPTIME_FILE

  live=""; dead=""; msg2=""; recheck=true
  for addr in $HOSTS; do
   live=0
   if ! do_host_remote_cmd $addr "uptime"; then
    live=`grep "^$addr" $UPTIME_FILE.old | cut -f2 -d' '`
    if [ "$live" = "" ]; then
     live=0
    fi
   fi
   # TUTAJ DZIELIMY $live NA POLA

   # pierwsze pole to uptime w sekundach
   echo "$addr `echo "$live" | head -n1 | awk '{printf("%s", $1)}'`" >> $UPTIME_FILE

   # kolejne linie to "status alert"
   live=`echo "$live" | sed -e "1d"`
   if [ "$live" != "" ]; then
    msg2="$msg2$addr:\n$live\n"
    msg2="$msg2*****************************************************************\n\n"
   fi
  done

  # sortujemy, zestawiamy i porownujemy nowy wynik z poprzednim
  sort -f -k1 $UPTIME_FILE > $UPTIME_FILE.new
  msg=`join -j1 $UPTIME_FILE.new $UPTIME_FILE.old 2>/dev/null | awk '
   $2==0 {next}
   $2!=0 && $3==0 {print " * " $1 " (old uptime is 0)"; next}
   $2 < $3        {print " * " $1}
  '`

  if [ "$msg" != "" -o "$dead" != "" -o "$msg2" != "" ]; then
   [ "$msg" != "" ]  && msg="Next $SYSTEM_NAME hosts are reboot:\n$msg\n\n"
   [ "$msg2" != "" ] && msg="${msg}Next $SYSTEM_NAME hosts are status alert:\n$msg2\n\n"
```

```
    [ "$dead" != "" ] && msg="${msg}We couldn't test next $SYSTEM_NAME host (SSH problem):⤸
    \n$dead"
    send_info "$SYSTEM_NAME: host reboot/status alert" "$msg"
   fi
  ;;

# SPRAWDZANIE STANU HOSTÓW (zajętość dysków, smart, logi)
 "status"|"raport")
  live=""; dead=""; recheck=true
  msg=""
  for addr in $HOSTS; do
   if do_host_remote_cmd $addr "$CMD"; then
    if [ "$live" != "" ]; then
     msg="$msg$addr:\n$live\n"
     if [ $CMD != "raport" ]; then
      msg="$msg****************************************************************\n\n"
     fi
    fi
   fi
  done

  if [ $CMD = "raport" ]; then
   title="raport"
   msg="$msg\nNext host are ping only:\n$HOSTS_PING_ONLY"
   msg="$msg\nNext host are in $BLOCK_FILE:\n`while read f; do echo " * $f"; done < ⤸
   $BLOCK_FILE`"
  else
   title="status alert"
  fi

  if [ "$msg" != "" -o "$dead" != "" ]; then
   [ "$msg" != "" ]  && msg="Next $SYSTEM_NAME hosts are $title:\n\n$msg\n\n"
   [ "$dead" != "" ] && msg="${msg}We couldn't test next $SYSTEM_NAME host (SSH problem):⤸
   \n$dead"
   send_info "$SYSTEM_NAME: hosts $title" "$msg"
  fi
 ;;

# INSTALACJA / AKTUALIZACJA SYSTEMU
 # połączenia bazują na ssh użytkownika wywołującego
 # $2, $2, ... - lista hostów
 "install"|"update")
  for addr in $@; do
   echo $addr

   [ "$CMD" = "install" ] && echo -e "\\033[34minstaluje SYSMON na $addr\\033[0m" && ssh ⤸
   $addr '
    cat <<EOF > $HOME/sysmon-install.sh
#!/bin/bash
[ "`uname`" = "SunOS" ] && (/usr/sbin/useradd -c "System Monitoring script" -d ⤸
"'"$system_dir"'" -m -s "/bin/bash" sysmon || echo "ADD sysmon USER ERROR at SunOS")
[ "`uname`" = "Linux" ] && (adduser --system --disabled-password --gecos "System Monitoring ⤸
script" --home "'"$system_dir"'" --shell "/bin/bash" sysmon || echo "ADD sysmon USER ERROR at ⤸
Linux")
mkdir -p '"$system_dir"'/.ssh/
echo "command=\"'"$system_dir"'/sysmon_client.sh\",no-port-forwarding,no-agent-forwarding,no-⤸
X11-forwarding,no-pty '"`cat $system_dir/.ssh/id_rsa.pub`"'" > '"$system_dir"'/.⤸
ssh/authorized_keys
chown -R sysmon:nogroup '"$system_dir"'
exit 0
```

```
EOF
    chmod +x $HOME/sysmon-install.sh
    if which pfexec >/dev/null; then
     pfexec $HOME/sysmon-install.sh || sudo $HOME/sysmon-install.sh
    else
     sudo $HOME/sysmon-install.sh
    fi
    rm $HOME/sysmon-install.sh
   '
   echo -e "\\033[34mkopiuje pliki SYSMONa na $addr\\033[0m"
   scp -p $system_dir/sysmon_server.* $system_dir/sysmon_client.sh $addr:
   echo -e "\\033[34mumieszczam  pliki SYSMONa w $addr:$system_dir\\033[0m"
   ssh $addr 'pfexec cp -p $HOME/sysmon_* '"$system_dir"' || sudo cp -p $HOME/sysmon_* ⤸
   '"$system_dir"
  done
  echo "dla dzialania kontroli S.M.A.R.T oraz temperatury wymagane jest zainstalowanie ⤸
  odpowiednich narzedzi (hd w Solarisach, smartctl w Linuxach, ipmitool)
  oraz dodanie do /etc/sudoers odpowiednio
  sysmon  ALL =NOPASSWD:/usr/bin/hd, /usr/bin/ipmitool
  sysmon  ALL =NOPASSWD:/usr/sbin/smartctl, /usr/bin/ipmitool"
;;
 "fast-update")
  for addr in ${@-$HOSTS}; do
   echo $addr

   scp -p $system_dir/sysmon_server.* $system_dir/sysmon_client.sh $addr:$system_dir/
  done
 ;;
esac

# usuniecie tymczasowego pliku stanowiącego bufor błędów
rm $ERROR_TMP_FILE 2>/dev/null

# usuniecie pliku blokady uruchomienia
rm $RUN_FILE
```

SYSMON_CLIENT.SH

```
#!/bin/bash


### KONFIGURACJA ###
# sciezka do katalogu skryptu
system_dir="/var/lib/sysmon"

# lista numerów atrybutów monitorowanych przez S.M.A.R.T
smart_attrib_list="5|10|196|197|198"

# wlaczenie reszty konfiguracji zależnej od hosta
# możliwe ustawienia:
#     not_run_smart="true" # wyłączenie testowania S.M.A.R.T
#     not_run_temp="true" # wyłączenie testowania temperatury
# możliwe jest także nadpisanie ustawień podanych powyżej
if [ -e $system_dir/sysmon_client.conf ]; then
 source $system_dir/sysmon_client.conf
fi

raport() {
```

```
 all_OK=true;
 if [ "$not_run_temp" = "true" -o ! -x /usr/bin/ipmitool ]; then
  echo " * Temperature monitoring is stopped"
  all_OK=false;
 fi
 if [ "$not_run_smart" = "true" ] || [ $UNAME = "SunOS" -a ! -x /usr/bin/hd ] || [ $UNAME = ⤸
 "Linux" -a ! -x /usr/sbin/smartctl ]; then
  echo " * S.M.A.R.T monitoring is stopped"
  all_OK=false;
 fi
 if $all_OK; then
  echo " * All tests is ON"
fi
}

uptime_sek() {
 # zwracamy uptime w postaci liczby sekund
 case $UNAME in
  "SunOS")
   LANG=C uptime | tr -d ',' | nawk '
    t2s(t) {
     split(t, tmp, ":");
     if (2 in tmp) return tmp[1]*60*60+tmp[2]*60
     return tmp[1]*6
    }
    $4~/day/ {
     printf $3*60*60*24 + t2s($5);
     next
    }
    {
     printf t2s($3)
    }
   '
  ;;
  "Linux")
   cut -f1 -d' ' /proc/uptime
  ;;
 esac
}

temp() {
 if [ "$not_run_temp" = "true" -o ! -x /usr/bin/ipmitool ]; then
  return
 fi
 sudo ipmitool -I open sdr | awk '
  BEGIN {CPU=0; PLANAR=0; AMBIENT=0}
  $1=="CPU" && $3=="Temp" && $5>CPU {CPU=$5}
  $1~"cpu..dietemp" && $3>CPU {CPU=$3}
  $1=="Planar" && $2=="Temp" && $5>PLANAR {PLANAR=$5}
  $1=="Ambient" && $2~"Temp" && $4>AMBIENT {AMBIENT=$4}
  $1=="ambienttemp" && $3>AMBIENT {AMBIENT=$3}
  END {if (CPU>'${CPU_TEMP_ALERT-68}' || PLANAR>'${PLANAR_TEMP_ALERT-30}' || AMBIENT>'${⤸
  AMBIENT_TEMP_ALERT-40}')
   printf("High Temp Warning (cpu=%s planar=%s ambient=%s)\n", CPU, PLANAR, AMBIENT)}
 '
}

freespace() {
 if [ "$UNAME" = "Linux" ]; then
  df_opt="-P"
```

```
fi

# procentowo
 # msg=$(df -h $df_opt | nawk 'BEGIN{ret=0} NR>1 && $5+0>95 {print $0; ret=10} END{exit ret}')

# bezwzglednie
 msg=$(df -k $df_opt | nawk 'BEGIN{ret=0} $3+0>100000 && $4+0<100000 {print $0; ret=10} END{
 exit ret}')

if [ $? = 10 ]; then echo "DISK FREE SPACE WARNING:"; echo "$msg"; fi
}

watch_dmesg() {
 case $UNAME in
  "SunOS")
   msg=$(dmesg | egrep "`date | nawk '{printf("(%s %2d)|(%s %2d)", $2, $3, $2, $3-1) }'`" |
    egrep '(file system full)|(WARN)|(ERR)|(CRIT)|(ALERT)|(warning)|(error)|(alert)|(critical)
    ' |
    cut -b 17- | sort | uniq
   )
  ;;
  "Linux")
   msg=$(dmesg | tr -d '[]' |
    awk -v time="$(( `cut -f1 -d'.' /proc/uptime` - 3600*48 ))" '$1>time{$1=""; print $0}' |
    egrep '(WARN)|(ERR)|(CRIT)|(ALERT)|(warning)|(error)|(alert)|(critical)' | sort | uniq
   )
  ;;
 esac

if [ "$msg" != "" ]; then
  echo "DMESG WARNINGS:"
  echo "$msg"
fi
}

smart() {
 if [ "$not_run_smart" = "true" ]; then
  return
 fi

case $UNAME in
  "SunOS")
   if [ ! -x /usr/bin/hd ]; then
    return
   fi

   sudo hd -r | awk '
    /^[0-9]+ c[0-9]+t[0-9]+$/ {DYSK=$2; next}
    /^ *('"$smart_attrib_list"') / {printf("%s : %s\n", DYSK, $0)}
   ' > $system_dir/smart.new
  ;;
  "Linux")
   if [ ! -x /usr/sbin/smartctl ]; then
    return
   fi

   for d in /dev/sd[a-d] /dev/hd[a-d]; do sudo smartctl -A $d | awk -v DYSK=`basename $d` '
    /^ *('"$smart_attrib_list"') / {printf("%s : %s\n", DYSK, $0)}
   '; done > $system_dir/smart.new
  ;;
```

```
 esac

touch $system_dir/smart.old

if ! diff_out=`diff $system_dir/smart.old $system_dir/smart.new`; then
  echo "S.M.A.R.T. ATTRIBUTES CHANGE TO:"
  echo "$diff_out" | grep '^>' | cut -b2-

  bck_file=$system_dir/smart.`date "+%FT%T"`
  mv $system_dir/smart.old $bck_file
  mv $system_dir/smart.new $system_dir/smart.old

  echo "old attributes save to: $bck_file"
  echo "more info about S.M.A.R.T. attributes - http://en.wikipedia.org/wiki/S.M.A.R.T."
fi
}


# ustalenie komendy
cmd=$1
if [ "$cmd" = "" ]; then
 read cmd
fi


UNAME=`uname`
export LANG=C

case $cmd in
 "uptime")
  echo "`uptime_sek`"
  temp
;;
 "status")
  freespace
  watch_dmesg
  smart
;;
 "raport")
  raport
;;
esac
```

SYSMON_SERVER.CONF

```
# czy uzywamy pliku blokad zapobiegajacego wysylaniu wielokrotnych
# komunikatow o tym samym hoscie
USE_BLOCK=true

# surfix dla kopi bliku blokad gdy host ozyl ...
# puste jezeli nie chcemy robic backupu
BLOCK_BACKUP_SURFIX=".`date +'%FT%T'`"

# czy pomijamy hosty z pliku blokad w opołączeniach SSH
USE_BLOCK_FOR_SSH=true

# plik blokad
BLOCK_FILE="$system_dir/block.txt"
```

```
# ile razy prubujemy zanim uznamy za martwego
RECHECK_RETRY=2

# ile czasu [s] odczekujemy przy ponownych probach
RECHECK_TIME=30

# czy rechek robic nawet gdy dla poprzedniego hostu nie przyniusl skutku
RECHECK_ANY=false;



# nazwa monitorowanej grupy hostow uzywana w powiadomieniach
SYSTEM_NAME="ICM"

# konfiguracja adresow powiadamiania
ADMIN_SMTP=""
ADMIN_XMPP=""

# konfiguracja programow wysylajacych
SENDEMAIL_OPTIONS="-s localhost"
MAIL_FROM=""
SENDXMPP_AUTH_FILE="$system_dir/xmpp.conf"



# funkcja odpowiedzialna za rzeczywiste wykonanie komendy na zdalnym hoście
# $1 - host address, $2 - komenda
RUN_REMOTE() {
 echo "$2" | ssh -i $system_dir/sysmon.ssh_id $1
}



# konfiguracja skladowania informacji o uptime
UPTIME_FILE="$system_dir/uptime"

# plik tymczasowy przetrzymujący błędy przed ich wysłaniem ...
ERROR_TMP_FILE="$system_dir/error"


# konfiguracja monitorowanych hostow
HOSTS=""

#	lista pozostalych hostow
HOSTS_PING_ONLY=""


# lista urli do sprawdzania
URL_CHECK_LIST=""

# lista usług do sprawdzania przez dostępność portu
SERVICES_CHECK_LIST=""

# czy niedostępność hosta świadczoącego usługę jest błędem usługi (1 = tak, 0 = ⤸
nie)
CHECK_FOR_DEAD_SERVICES=0
```

```
# czy uzywamy pliku blokad dla sprawdzania usług
USE_BLOCK_FOR_SERVICES=true

# funkcja w ramach której można definiować własne testy usług
# (np. sprawdzanie odpowiedzi DNSowej przy pomocy 'dig @host pytanie')
# jeżeli wyprodukuje cokolwiek na STDOUT zsostanie to przesłane w ramach komunikatu o ⤸
stanie usług
# addional_services_check() { : }
```

SERVICES_CHECKER.SH

```
# testowany host
HOST=dragon.icm.edu.pl

# sprawdzana domena i typ rekordu
DNS_NAME=opcode.eu.org
DNS_TYPE=MX

# sprawdzane porty usług publicznych
PUB_PORTS=25,80,5060,5269
PUB_PORTS=25,80

# sprawdzane porty usług prywatnych
PRIV_PORTS=22,465,993,5222
PRIV_PORTS=22,465,993


LOCK_FILE=/tmp/$USER.hostchecker.state

[ -f $LOCK_FILE ] && read LEVEL RUN PID < $LOCK_FILE

if [ "$LEVEL" = "" ]; then
 LEVEL=0
 RUN=0
elif [ $LEVEL -eq 999 ]; then
 RUN=$(( $RUN + 1 ))
 if [ $RUN -gt 5 ]; then
  echo "$0 work too long !!!"
  kill -9 $PID
  LEVEL=0
 else
  echo "999 $RUN $PID" > $LOCK_FILE
  exit
 fi
else
 LEVEL=$(( $LEVEL + 1 ))
 if [ $LEVEL -gt 7 ]; then
  LEVEL=0
 fi
fi

echo "0 $RUN $$" > $LOCK_FILE


case $(( $LEVEL % 4 )) in
 0)
```

```
  DNS_OPT="-6 +notcp"
  NMAP_OPT="-6"
  ;;
 1)
  DNS_OPT="-4 +notcp"
  NMAP_OPT=""
  ;;
 2)
  DNS_OPT="-6 +tcp"
  NMAP_OPT="-6"
  ;;
 3)
  DNS_OPT="-4 +tcp"
  NMAP_OPT=""
  ;;
esac

# check DNS
if ! dig @$HOST $DNS_OPT +tries=0 +retry=0 +time=1 $DNS_NAME $DNS_TYPE | grep -v '^;' | grep ⤸
$DNS_TYPE > /dev/null; then
 echo "DNS check faild for HOST=$HOST DNS_NAME=$DNS_NAME DNS_TYP=$DNS_TYP"
fi

# check PORTS
nmap $NMAP_OPT -sT -p $PUB_PORTS -oN - $HOST | egrep '^[0-9]*/(tcp|udp)' | grep -v open

if [ $LEVEL -lt 2 ]; then
 nmap $NMAP_OPT -sT -p $PRIV_PORTS -oN - $HOST | egrep '^[0-9]*/(tcp|udp)' | grep -v open
fi

echo "$LEVEL $RUN $$" > $LOCK_FILE
```

Inteligentny Dom

Dział ten gromadzi projekty elektroniczno-programistyczne związane z automatyką domową - systemami sterowania urządzeniami elektrycznymi, kontroli dostępu, detekcji pożaru i włamania. Ewolucja stosowanych tutaj rozwiązań projektowych i komunikacyjnych (sieć RS485) opisana jest w ewolucja koncepcji projektów inteligentnego domu. Projekty czysto programistyczne związane z monitoringiem pracy różnych systemów (zarówno informatycznych jak i infrastrukturalnych) przedstawione są w dziale monitoring.

Głównymi elementami prezentowanego systemu są: centralny system zasilania DC i główny układ programowalnego kontrolera. Wyposażony jest on w kilka magistral RS485[1980]/ModbusRTU[1981] (do podłączania zdalnych modułów), I2C[1982] (do bezpośredniej realizacji wejść i wyjść) oraz port Ethernetu zapewniający komunikację TCP/IP. Oparty jest na mikrokontrolerze ARM[1983] z systemem Linux i bardziej szczegółowo opisany w centralka (nie tylko) alarmowa.

Ogólne założenia, wskazówki, itd. dotyczące projektowania obiektów z myślą o inteligentnych instalacjach, projektowania samych instalacji, punktów dystrybucyjnych a także całych serwerowni opisane są w instalacje i budynki.

Moduły wchodzące w skład poszczególnych projektów były tak tworzone aby dało się je łatwo wykorzystać w innych projektach, jako swego rodzaju klocki realizujące typowe funkcje związane z systemami sterowania obiektem:

- moduły kontrolne:
  - master systemu oparty na mikrokontrolerze ARM z systemem Linux, komunikatcją TCP/IP over Ethernet, i2c, rs485 - wykorzystane standardowe płytki tego typu, więcej szczegółów w centralka (nie tylko) alarmowa
- moduły z komunikacją Modbus/RS485:

- sterownik oparty na mikrokontrolerze Atmega8 z komunikacją poprzez interfejs RS485 - biblioteczka dla uC, wykorzystywany m.in. w:
  - manipulator do systemu alarmowego - wyświetlacz LCD i klawiatura z komunikacją MODBUS
  - sterownik oświetlenia - sterownik 4 linii PWM z niezależna detekcją zera dla każdej linii oraz 4 linii ON/OFF
  - rozproszone DI na magistrali Modbus/RS485 - może być bezproblemowo zrealizowane w oparciu o dev-board modułu podstawowego
- moduły wejśc/wyjść z komunikacją I2C:
  - moduły sterujące liniami DC:
    - moduły tranzystorowe do magistrali I2C - sterownik LED RGB / sterownik linii DC
  - moduł obsługi magistrali 1wire - centralka (nie tylko) alarmowa
  - moduł obsługi wejść z linii 3 stanowych, analogowych, cyfrowych z detekcją zmiany, … - centralka (nie tylko) alarmowa
- moduły wykonawcze sterowane sygnałem cyfrowym:
  - moduły sterujące liniami AC - triakowy sterownik obwodów 230V AC:
    - moduł triakowy ON/OFF
    - moduł triakowy z regulacją mocy PWM
- system zasilania DC

Centralka (nie tylko) alarmowa

Centralka oparta jest na układzie z procesorem ARM, systemem linux i interfejsem ethernetowym (raspberry-pi[1984]) i peryferiach podłączanych z wykorzystaniem i2c i portów szeregowych (jeżeli w danym module ich brakuje można ich ilość zwiększyć z użyciem układu FT4232 np. w formie modułu MMusb4232H[1985]). Zapewnia to modularność, elastyczność, łatwość w modyfikacji i rozbudowie takiej konstrukcji, a w przeciwieństwie do stosowanej w poprzedniej wersji magistrali równoległej nie wywołuje aż tak strasznej plątaniny połączeń.

Założeniem jest bezpośrednia (poprzez i2c) obsługa przez sterownik linuxowy linii wejściowych z czujników alarmowych (analogowych, gdyż brakuje otwartego standardu komunikacyjnego dla czujników adresowalnych, stosowane są wewnętrzne protokoły danego producenta systemu co uniemożliwia ich wykorzystanie w takim projekcie) - zarówno włamaniowych (np. PIR) jak i p.poż., a także części linii wejściowych i wyjściowych ogólnego przeznaczenia. Z wykorzystaniem magistral szeregowych RS485 i protokołu modbus-rtu podłączane są moduły rozproszonego DI (klawiatury i przełączniki oświetlenia z własnym uC oparte na Atmega8 i FreeMODBUS[1986]), sterowniki wykonawcze oświetlenia, gniazd i ogrzewania (moduły triakowe, moduły tranzystorowe do przekaźników z własnym uC oparte na Atmega8 i FreeMODBUS, czujniki oraz mierniki innych producentów używające modbusa (np. analizatory PM710).

Podłączanie czujników alarmowych

Czujniki systemów alarmowych podłączane jako rezystancyjne linie trójstanowe (alarm, sabotaż/awaria, normalny - schemat ideowy[1987](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/czujki_alarmowe.pdf`)). Bardzo istotnym zagadnieniem jest zapewnienie niezależności zasilania takich czujników od systemu głównego (zwarcie w linii zasilającej czujnik nie może powodować utraty zasilania przez centralkę). Podobnie linie te powinny być zabezpieczone przepięciowo (przed podaniem zbyt dużego napięcia), a linie sygnałowe powinny być optoizolowane - podanie tam zbyt dużego napięcia nie może uszkodzić lub nawet wyłączyć centralki.

Podłączenie rezystancyjne czujników (przy zachowaniu wyżej opisanych zabezpieczeń) utrudnia dokonanie sabotażu takiej linii w stosunku co do zwykłej linii DI (którą wystarczy przeciąć lub zewrzeć), ale jest on nadal możliwy - wystarczy podłączyć do linii sygnałowej źródło napięcia takiego jakie jest związane z stanem normalnym tej linii, wydawałoby się że problem rozwiąże zastosowanie losowo zmiennego napięcia, ale to też da się obejść. Dodatkowym problemem jest stan przekaźnika w momencie gdy czujka nie ma zasilania - powinien on odpowiadać stanowi usterki, w ostateczności alarmu, ale nigdy stanowi normalnemu linii, tak aby nie dało się obejść systemu przez odłączenie zasilania (lub masy) od czujki. Jedynym rozwiązaniem dającym dużą pewność że odczytywany sygnał jest faktycznie od czujki jest zastosowanie szyfrowanej komunikacji protokolarnej np. poprzez rs485 lub ethernet - wymaga to stosowania zabezpieczonego przed dostępem modułu elektronicznego umieszczanego w samej czujce. Niezależnie od tego pozostaje kwestia traktowania stanu usterki w zależności od tego w jakim jest to systemie (p.poż czy włamaniowym) oraz czy system jest uzbrojony czy rozbrojony.

Pierwotnie planowana była realizacja detekcja stanu linii trójstanowej poprzez zestaw dwóch komparatorów

okienkowych, wytwarzających wytwarza dwa sygnały cyfrowe (alarm i usterka). Zaprojektowana została realizacja takiego układu wraz z systemem detekcji zmiany i zatrzaskiwania informacji o niej (wytwarzanie sygnału INT) oparta na elementach o małym stopniu integracji - moduł dla pojedynczej linii (schemat ideowy[1988](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.pdf`) i projekt PCB[1989](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.pcb.pdf`)) oraz backplane dla 4 takich modułów (schemat ideowy[1990](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.pdf`) i projekt PCB[1991](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.pcb.pdf`)). Niestety rozwiązanie to jest dość skomplikowane i bardzo duże.

Próba znalezienia układu scalonego bufora który potrafiłby wykrywać zamianę i zatrzaskiwać informację o niej doprowadziła do znalezienia MCP23008. Zgodnie z dokumentacją układu MCP23008 powinien on być wstanie odnotowywać informacje o liniach wywołujących przerwania w trakcie trwania przerwania, dzięki czemu mógłby posłużyć do realizacji takiej detekcji zmiany znacznie upraszczając ten układ, jednak przeprowadzone testy wskazują że niestety nie działa on w ten sposób. Funkcjonalność taką można realizować też na ADC lub poprzez (odpowiednio szybkie) czytanie sygnałów wychodzących z komparatorów okienkowych (alarm i usterka) i programową ich analizę, jednak tracimy wtedy przewagę szybkości realizacji sprzętowej. Mimo to ostatecznie zostało to zrealizowane z wykorzystaniem przetwornika analogowo-cyfrowego wbudowanego w Atmega8.

Realizacja centralki - moduły

Wstępne założenia funkcjonalne (jedna szyna i2c -> max. 16 układów) wyglądały:

- multiplekser magistrali i2c oparty na PCA9544 lub PCA9548
- płytka analog + 1wire
  - optoizolacja I2C na bazie P82B96 + układ zasilania części izolowanej (czyli prawie całej płytki i magistral 1wire)
  - 4x lub 8x magistrala 1wire (1x DS2482S-100 + switch 1wire oparty na 4052 lub 4051)
  - 8x analog-input (2x PCF8591)
  - 2x rs485 (2x SC16IS760IPW lub 1x SC16IS762IPW + optoizolacja + driver rs485)
- płytki (jeden z bitów adresowych ustawiany zworką) z zestawem wejść:
  - 32x digital-input (4x MCP23008 + 8x transoptor ISP620-4X, ISP621-4X lub inny)

Ponaddto planowany był moduł z mikrokontrolerem do obsługi czujnika wilgotności SYH-1S.

Ostatecznie została ona zrealizowana w oparciu o raspberry-pi z podłączonym po USB układem FT4232H (realizującym poprzez odpowiednie interfejsy linie rs485/modbus - schemat ideowy[1992](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/ft4232-rs485.pdf`), projekt PCB[1993](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/ft4232-rs485.pcb.pdf`)) oraz podłączonymi po i2c modułami:

- wyjść NPN - opartym na MCP23008 i ULN2803APG (patrz sterownik LED) do sterowania przekaźników wykonawczych
- linii 1wire - opartym na DS2482 i multiplekserze analogowym sterowanym poprzez MCP23008 (schemat ideowy[1994](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/i2c-1wire.pdf`), projekt PCB[1995](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/i2c-1wire.pcb.pdf`))
- wejść wyjść cyfrowych i wejść analogowych opartym na mikrokontrolerze Atmega8 (schemat ideowy[1996](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/atmega8_i2c-slave.pdf`), projekt PCB[1997](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/atmega8_i2c-slave.pcb.pdf`), program sterujący)

Skrypty sterujące dla raspberry-pi:

- run.sh - głowny plik skryptu
- output.sh - obsługa sygnalizacji i sterowanych urządzeń
- input.sh - obsługa wejścia

Ponaddto zamieszczam przydatne pliki, skrypty dla raspberry-pi

- rpi-config.txt[1998] - konfig dla raspberry-pi uruchamiający go z wyjściem HDMI w full-hd nawet gdy nie ma podłączonego monitora
- rpi-rc.local[1999] - plik startowy kontrolujący poprawność uruchomienia sieci i odpalający access point wifi

— koncepcja_odczytu_di.c[2000] - rozważania koncepcyjne na temat odczytu DI w C

Pobierz[2001]

ATMEGA8_I2C-SLAVE.C

```
/*
 * program pozwlający na wykorzystanie uC Atmega8 (a także wielu innych uC z rodziny AVR)
 * w roli inteligentnego bufora I/O obsługującego:
 *     16 linii cyfrowych wejścia wyjścia (port B i D uC)
 *     4 linii z możliwością wyboru pomiędzy cyfrowym wejściem/wyjściem, analogowym ⤸
wejściem,
 *                    rezystancyjnym wejściem trójstanowym (piny 0..3 portu C uC)
 *
 *
 * kompilacja i wgranie przez ISP z użyciem USBasp:
 *   avr-gcc -mmcu=atmega8 -Os -o sterownik.o main.c
 *   avr-objcopy -O ihex sterownik.o sterownik.hex
 *   sudo avrdude -c usbasp -p m8 -U flash:w:sterownik.hex
 *
 * konfiguracja układu na wewnętrzny szybki oscylator
 *   sudo avrdude -c usbasp -p m8 -U lfuse:w:0xe4:m
 *
 *
 * układ zapewnia m.in:
 *     - programową eliminację drgań (badanie stabilności stanu) na wejściach cyfrowych
 *     - detekcję zmiany stanu wraz z zatrzaskiwaniem takiej informacji
 *        (umożliwia wykrycie faktu zmiany stanu pinu pomimo iż pomiędzu odczytami i2c
 *        zdążył on wrócić do pierwotnego stanu)
 *
 * całość konfiguracji układu (kirunki transmisji na poszczególnych pinach DIO,
 * wartości progowe dla linii trójstanowych, wykorzystanie pinów portu C jako
 * ADC lub DIO, ...) odbywa się poprzez szyne I2C
 *
 * rejestry I2C:
 *   ro 0x00          – general status
 *   ro 0x40 - 0x59 – stabilność pinu
 *
 *   rw 0x60 - 0x78 –   ustawienie ilości okresów sprawdzania stabilności potrzebnych do ⤸
uznania pinu za stabilny
 *   rw 0x79          –   ustawienie ilości tyknięć timera w jednym okresie sprawdzania ⤸
stabilności
 *   rw 0x7A          –   ustawienie włączenia i prędkości sprawdzania stabilności:
 *                          0x00 - wyłączony
 *                          0xFn - włączony, dla n=2 => speed=F_CPU/8     n=3 => speed=F_CPU/64     ⤸
n=4 => speed=F_CPU/256
 *   rw 0x7B          – maska ustawiająca czytane poprty i porty z sprawdzaniem stabilności
 *   rw 0x80 - 0x83 – ustawienie limitów linii trójstanowych - normal minimum
 *   rw 0x84 - 0x87 – ustawienie limitów linii trójstanowych - normal maximum
 *   rw 0x88 - 0xFB – ustawienie limitów linii trójstanowych - active minimum
 *   rw 0x8C - 0x8F – ustawienie limitów linii trójstanowych - active maximum
 *   rw 0x91 - 0x93 – ustawienie wartości PORTn    n=B,C,D
 *   rw 0x95 - 0x97 – ustawienie wartości DDRn    n=B,C,D (stan wysoki - output)
 *   rw 0x99 - 0x9B – wartość PINn    n=B,C,D
 *   ro 0XA0 - 0xA3 – wartość z konwersji anlog-digital z odpowiedniej linii
 *   rw 0xAF          – ustawienie włączenia i prędkości ADC:
 *                        0x00 - wyłączony
 *                        0x0n - włączony, dla n=3 => speed=F_CPU/8     ...     n=6 => speed=F_CPU/64
 *   rw 0xBn          – bajt n (znaczenia bajtów opisane poniżej w define) statusu portu B
```

```
 *    rw 0xCn              – bajt n (znaczenia bajtów opisane poniżej w define) statusu portu C
 *    rw 0xDn              – bajt n (znaczenia bajtów opisane poniżej w define) statusu portu D
 *    rw 0xE0              – status linii trójstanowych
 *
 *    Przykłady:
 *      # dioda na porcie D
 *      i2cset -y $BUS_ADDR 0x10 0x97 0x20
 *      alias l_on='i2cset -y $BUS_ADDR 0x10 0x93 0x20'
 *      alias l_off='i2cset -y $BUS_ADDR 0x10 0x93 0x00'
 *
 *      # odczyt portu D z podciąganiem
 *      i2cset -y $BUS_ADDR 0x10 0x93 0xff
 *      for i in `seq 1 30`; do sleep 0.1; i2cget -y $BUS_ADDR 0x10 0x9B; done
 *
 *      # buforowany odczyt portu D
 *      i2cset -y $BUS_ADDR 0x10 0x7a 0xff
 *
 *      i2cget -y $BUS_ADDR 0x10 0xD0
 *
 *      i2cget -y $BUS_ADDR 0x10 0xD2
 *      i2cget -y $BUS_ADDR 0x10 0xD2
 *      i2cget -y $BUS_ADDR 0x10 0xD2
 *
 *
 *      # odczyt ADC(2)
 *      i2cget -y $BUS_ADDR 0x10 0xAf 0xf3
 *      i2cset -y $BUS_ADDR 0x10 0xAf 0xf3;    i2cget -y $BUS_ADDR 0x10 0xA2;
 *      # tu jest jakiś dziwny problem objawiający się koniecznościa każdorazowego
 *      # ponawiania ustawienia konfiguracji ADC ... ale niech będzie ...
 */

#include <inttypes.h>

#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>

#define I2C_ADDRESS 0x10


/// bit mask of last port value
#define _VALUE_                         0
/// bit mask of input stability
#define _IS_STABLE_                     1

/// bit mask of pins that changed values
#define _CHANGE_MASK_                   2
/// bit mask of pins values that caused set bits in _CHANGE_MASK_
#define _CHANGE_VALUE_                  3
/// bit mask of pins that changed values to level specified in _CHANGE_TO_LEVEL_VAL_
#define _CHANGE_TO_LEVEL_MASK_          4

/// bit mask of pins that changed stable values
#define _STABLE_CHANGE_MASK_            5
/// bit mask of pins values that caused set bits in _STABLE_CHANGE_MASK_
#define _STABLE_CHANGE_VALUE_           6
/// bit mask of pins that changed stable values to level specified in _CHANGE_TO_LEVEL_VAL_
#define _STABLE_CHANGE_TO_LEVEL_MASK_   7
```

```
/// bit mask for determinate pins using usnatble value of input to settings _VALUE_, ⤸
_CHANGE_*_
#define _USE_UNSTABLE_VALUE_          12
/// bit mask for modify status on change pin value based on _CHANGE_MASK_
#define _INFO_ON_CHANGE_              13
/// bit mask for modify status on change pin value based on _CHANGE_TO_LEVEL_MASK_
#define _INFO_ON_CHANGE_TO_LEVEL_     14
/// bit mask determinate pins value for _CHANGE_TO_LEVEL_MASK_
#define _CHANGE_TO_LEVEL_VAL_         15


volatile uint8_t status_mask;
volatile uint8_t status_b[16];
volatile uint8_t status_c[16];
volatile uint8_t status_d[16];
volatile uint8_t status_tri[16];
volatile uint8_t status_adc_val[4];
volatile uint8_t status_input_mask, status_adc_mode;
volatile uint8_t adc_normal_max[4], adc_normal_min[4], adc_active_max[4], adc_active_min[4];
volatile uint8_t pin_stability[24], pin_stability_init[24];


/// UPDATE STATUS INFO, AFTER READ PORT VALUE
inline void __attribute__((always_inline)) check_port_value(uint8_t nval, volatile uint8_t ⤸
*status, uint8_t num) {
 uint8_t nmask, i;

nmask = nval ^ status[_VALUE_];

// when port value is changed
 if (nmask != 0x00) {
  // set suitable pin(s) status as unstable
  if (status_input_mask & (0x10 << num)) {
   for (i=0; i<8; i++) {
    if (nmask & (1<<i)) {
     pin_stability[ i | (num << 3) ] = pin_stability_init[ i | (num << 3) ];
     status[_IS_STABLE_] &= ~(1<<i);
    }
   }
  }

  cli();
  // store new port value
  status[_VALUE_] = nval;
  // store values that caused the (first) change of pin
  status[_CHANGE_VALUE_] |= nmask & (~status[_CHANGE_MASK_]) & nval;
  // update info about changes (for pins witch info-on-change)
  status[_CHANGE_MASK_] |= nmask;
  // update info about changes (for pins witch info-on-change-to-level)
  status[_CHANGE_TO_LEVEL_MASK_] |= nmask & (~(nval ^ status[_CHANGE_TO_LEVEL_VAL_]));

  // update global status
  if (status[_USE_UNSTABLE_VALUE_] & (
    (status[_CHANGE_MASK_] & status[_INFO_ON_CHANGE_]) |
    (status[_CHANGE_TO_LEVEL_MASK_] & status[_INFO_ON_CHANGE_TO_LEVEL_])
   )) {
    status_mask |= 1 << num;
  }
  sei();
 }
```

```
}

/// UPDATE STATUS INFO, AFTER STABILIZE PIN VALUE
inline void __attribute__((always_inline)) set_pin_stable(uint8_t pin_num, volatile uint8_t ⤸
*status, uint8_t num) {
 uint8_t nval, nmask;

nmask = 1 << pin_num;
 nval = status[_VALUE_] & nmask;

// store values that caused the (first) change of pin
 status[_STABLE_CHANGE_VALUE_] |= (~status[_CHANGE_MASK_]) & nval;
 // update info about changes (for pins witch info-on-change)
 status[_STABLE_CHANGE_MASK_] |= nmask;
 // update info about changes (for pins witch info-on-change-to-level)
 status[_STABLE_CHANGE_TO_LEVEL_MASK_] |= nmask & (~(nval ^ status[_CHANGE_TO_LEVEL_VAL_]));

// set stable info
 status[_IS_STABLE_] |= nmask;

// update global status
 if ( (status[_STABLE_CHANGE_MASK_] & status[_INFO_ON_CHANGE_]) |
   (status[_STABLE_CHANGE_TO_LEVEL_MASK_] & status[_INFO_ON_CHANGE_TO_LEVEL_])
  ) {
   status_mask |= 1 << num;
 }
}

/// UPDATE STATUS INFO, AFTER READ TRI-STATE VALUE
inline void __attribute__((always_inline)) check_tristate_value(uint8_t nval, uint8_t chan) {
 if (nval != status_adc_val[chan]) {
  status_adc_val[chan] = nval;
  status_mask |= 0x10 << chan;

  if (nval < adc_normal_max[chan] && nval > adc_normal_min[chan]) { // NORMAL STATE
   if (((status_tri[_VALUE_] >> chan) & 0x11) != 0x00) {
    status_tri[_VALUE_] = status_tri[_VALUE_] & (0x11 << chan);
    if (! (status_tri[_CHANGE_MASK_] & (1 << chan))) { // first change
     status_tri[_CHANGE_MASK_] |= 1 << chan;
     status_mask |= 0x08;
    }
   }
  } else if (nval < adc_active_max[chan] && nval > adc_active_min[chan]) { // ACTIVE STATE
   if (((status_tri[_VALUE_] >> chan) & 0x11) != 0x01) {
    status_tri[_VALUE_] = (status_tri[_VALUE_] & (0x11 << chan)) | 0x01;
    if (! (status_tri[_CHANGE_MASK_] & (1 << chan))) { // first change
     status_tri[_CHANGE_MASK_] |= 1 << chan;
     status_tri[_CHANGE_VALUE_] |= 0x01 << chan;
     status_mask |= 0x08;
    }
   }
  } else { // ERROR STATE
   if (((status_tri[_VALUE_] >> chan) & 0x11) != 0x10) {
    status_tri[_VALUE_] = (status_tri[_VALUE_] & (0x11 << chan)) | 0x10;
    if (! (status_tri[_CHANGE_MASK_] & (1 << chan))) { // first change
     status_tri[_CHANGE_MASK_] |= 1 << chan;
     status_tri[_CHANGE_VALUE_] |= 0x10 << chan;
     status_mask |= 0x08;
    }
   }
```

```
  }
 }
}

/// CLOCK INTERRUPT - DE-BOUNCING
ISR(TIMER1_COMPA_vect, ISR_NOBLOCK) {
 uint8_t i;
#ifdef DEBUG
 PORTB |= _BV( PB4 );
#endif

if ((status_input_mask & 0x11) == 0x11) {
  for (i=0; i<8; i++) {
   if (pin_stability[i] > 0) {
    if (--pin_stability[i] == 0) {
     set_pin_stable(i, status_b, 0);
    }
   }
  }
 }
 if ((status_input_mask & 0x22) == 0x22) {
  for (; i<16; i++) { // TODO: potrzebujemy sprawdzać 4 górne bity portu C ?
   if (pin_stability[i] > 0) {
    if (--pin_stability[i] == 0) {
     set_pin_stable(i-8, status_c, 1);
    }
   }
  }
 }
 if ((status_input_mask & 0x44) == 0x44) {
  for (; i<24; i++) {
   if (pin_stability[i] > 0) {
    if (--pin_stability[i] == 0) {
     set_pin_stable(i-16, status_d, 2);
    }
   }
  }
 }

// reset timera
 TCNT1 = 0x0000;

#ifdef DEBUG
 PORTB &= ~( _BV( PB4 ) );
#endif
}

/// READ ADC DATA
uint8_t adc_chan = 0;
ISR(ADC_vect, ISR_NOBLOCK) {
 uint8_t val = ADCH;
 uint8_t chan = adc_chan;

#ifdef DEBUG
 PORTB |= _BV( PB3 );
#endif

// start next channel
 if (++adc_chan > 3)
  adc_chan = 0;
```

```
 ADMUX = adc_chan | _BV(ADLAR);
 ADCSRA = status_adc_mode;

// check value
 check_tristate_value(val, chan);

#ifdef DEBUG
 PORTB &= ~( _BV( PB3 ) );
#endif
}


/// MAIN() - INITIALIZING   AND   READ GPIO DATA IN MAIN LOOP
int main( void ) {
#ifdef DEBUG
 DDRB = 0x1c;
 PORTB = 0;
#endif

uint8_t i;

for (i=0; i<16; i++) {
  status_b[i] = 0;
  status_c[i] = 0;
  status_d[i] = 0;
 }
 for (i=12; i<14; i++) {
  status_b[i] = 0xff;
  status_c[i] = 0xff;
  status_d[i] = 0xff;
 }

for (i=0; i<24; i++) {
  pin_stability_init[i] = 10;
 }

// counter - przerwanie co 1/8000000 * 64 * 40 = 320us (przy F_CPU = 8MHz)
 TCNT1 = 0x0000;
 TIMSK |= _BV( OCIE1A ); // enable interrupt from this timer
 OCR1A = 40;
 TCCR1B = 0;
  // włączony (wraz z ustawieniem prędkości) zostanie stosownym poleceniem I2C

// konfiguracja I2C jako slave
 TWAR = I2C_ADDRESS << 1;
  // adres, nie reagujemy na generall call
 TWCR = _BV(TWEN) | _BV(TWIE) | _BV(TWINT) | _BV(TWEA);
  // włączony | włączone generowanie przerwań | wyczyszczona flaga przerwań | ⤸
  włączone generowanie ACK

// włączenie obsługi przerwań
 sei();

// konfiguracja ADC
 ADMUX = _BV(ADLAR);
  // ustawiamy na kanał 0 (PC0), wynik dosunięty do lewej, napięcie odniesienia z AREF
 ADCSRA = 0;
  // włączony (wraz z ustawieniem prędkości) zostanie stosownym poleceniem I2C
```

```
status_input_mask = 0xFF;

// pętla główna odczyt GPIO
 while(1) {
  if (status_input_mask & 0x01) check_port_value(PINB, status_b, 0);
  if (status_input_mask & 0x02) check_port_value(PINC, status_c, 1);
  if (status_input_mask & 0x04) check_port_value(PIND, status_d, 2);

#ifdef DEBUG
  //if (PORTB & _BV( PB5 ))     PORTB &= ~( _BV( PB5 ) );    else      PORTB |= ( _BV( PB5 ) );
  PORTB |= ( _BV( PB5 ) );
  PORTB &= ~( _BV( PB5 ) );
#endif
 }
}


/// RETURN (AND CLEAN IF NEED) STATUS INFO
inline uint8_t __attribute__((always_inline)) read_status(volatile uint8_t *status, uint8_t ⤸
index) {
 uint8_t ret;
 ret = status[index];
 if (index >= _CHANGE_MASK_ && index <= _STABLE_CHANGE_TO_LEVEL_MASK_)
  status[index] = 0;
 return ret;
}

/// WRITE STATUS INFO
inline void __attribute__((always_inline)) write_status(volatile uint8_t *status, uint8_t ⤸
index, uint8_t val) {
 if (index >= _USE_UNSTABLE_VALUE_ && index <= _CHANGE_TO_LEVEL_VAL_)
  status[index] = val;
}

/// TWI COMMUNICATION
uint8_t twi_cmd = 0;
ISR( TWI_vect ) {
 uint8_t status = TWSR & 0xf8;
 uint8_t mode;
 uint8_t ack = 1;

/*
 if (status == 0xA0) { // hack for fast servicing of repeated start
  TWCR = (1<<TWEN) | (1<<TWIE) | (1<<TWINT) | (1<<TWEA);
  return;
 }
 */

switch (status) {
  // RECIVER (i2c write mode)
  case 0x68:
  case 0x60: // otrzymano adres, zwrócono ACK,                    ==>> odbiór danych
   twi_cmd = 0xff;
   break;
  case 0x80: // otrzymano dane, zwrócono ACK                       ==>> odbiór danych (kolejny ⤸
  bajt)
   if (twi_cmd == 0xff) { // reseved for internal use
    twi_cmd = TWDR;
   } else {
    mode = twi_cmd & 0xF0;
```

```
switch (mode) {
 case 0x60: // INITIAL PIN STABILITY
 case 0x70: // (number of TIMER1_COMPA cycles to wait before considering the value of a ⤸
 stable)
  mode = twi_cmd & 0x1F;
  if (mode < 24) {
   pin_stability_init[mode] = TWDR;
  } else if (mode == 24) {
   // number of timer ticks in one TIMER1_COMPA cycle
   OCR1A = TWDR;
  } else if (mode == 25) {
   // de-bouncing status
   // 0xFn - enable de-bouncing timer interrupt with timer speed determinate by n
   //          (see TCCR1B register in Atmega8 documentation)
   //          n=2 => speed=F_CPU/8     n=3 => speed=F_CPU/64     n=4 => speed=F_CPU/256
   // 0x00 - disable de-bouncing timer
   mode = TWDR;
   if (mode & 0xF0) {
    TCCR1B = mode & 0x07;
   } else {
    TCCR1B = 0;
   }
  } else if (mode == 26) {
   // input port read / de-bouncing bit-mask:
   // 0x01 - reaad port B
   // 0x10 - de-bouncing port B
   // 0x02 - reaad port C
   // 0x20 - de-bouncing port C
   // 0x04 - reaad port D
   // 0x40 - de-bouncing port D
   status_input_mask = TWDR;
  }
  break;
 case 0x80: // TRI-STATE limits
  mode = twi_cmd & 0x0F;
  if (mode < 4)
   adc_normal_min[mode] = TWDR;
  else if (mode < 8)
   adc_normal_max[mode - 4] = TWDR;
  else if (mode < 12)
   adc_active_min[mode - 8] = TWDR;
  else if (mode < 16)
   adc_active_max[mode - 12] = TWDR;
  break;
 case 0x90:
  mode = twi_cmd & 0x0F;
  switch (mode) {
   // mode & 0x0c == 0b00   =>   PORTx (port x output value / port x pullup resistor mask)
   case 0x01:
    PORTB = TWDR;
    break;
   case 0x02:
    PORTC = TWDR;
    break;
   case 0x03:
    PORTD = TWDR;
    break;
   // mode & 0x0c == 0b01   =>   DDRx (port x direction)
   case 0x05:
    DDRB = TWDR;
```

```
        break;
      case 0x06:
        DDRC = TWDR;
        break;
      case 0x07:
        DDRD = TWDR;
        break;
      // mode & 0x0c == 0b10    =>    PINx (port x input value)
      case 0x09:
        PINB = TWDR;
        break;
      case 0x0a:
        PINC = TWDR;
        break;
      case 0x0b:
        PIND = TWDR;
        break;
    }
    break;
  case 0xA0: // ADC
    if (twi_cmd == 0xAF) {
      // 0xmn - enable ADC (and tri-state line) with speed determinate by n
      //           (see ADCSRA register in Atmega8 documentation)
      //           n=3 => speed=F_CPU/8    n=4 => speed=F_CPU/16    n=5 => speed=F_CPU/32    n=6 =>
      speed=F_CPU/64
      //           m - set first channel to read
      // 0x00 - disable ADC (and tri-state line)
      mode = TWDR;
      if (mode != 0) {
        // prędkość ADC na podstawie wartości odebranej w poleceniu włączenia
        status_adc_mode = _BV(ADEN) | _BV(ADSC) | _BV(ADIE)|_BV(ADIF) | mode & 0x07;
        // ADC włączony, pelecenie startu konwersji, włączone generowanie przerwań

        // kanał od którego zaczynamy
        adc_chan = mode & 0x30;
        ADMUX = adc_chan | _BV(ADLAR);
      } else {
        status_adc_mode = 0;
      }
      ADCSRA = status_adc_mode;
    }
  case 0xB0: // PORT B
    write_status(status_b,   twi_cmd & 0x0f,  TWDR);
    break;
  case 0xC0: // PORT C
    write_status(status_c,   twi_cmd & 0x0f,  TWDR);
    break;
  case 0xD0: // PORT D
    write_status(status_d,   twi_cmd & 0x0f,  TWDR);
    break;
  case 0xE0: // TRI-STATE
    write_status(status_tri, twi_cmd & 0x0f,  TWDR);
    break;
  }
  twi_cmd = 0x00;
  ack = 0;
}
break;
case 0x88: // otrzymano dane, zwrócono NOT ACK                ==>> oczekiwanie na adres
break;
```

```
case 0xA0:  // otrzymano STOP lub powtórzony START                   ==>> oczekiwanie na adres
 break;

// TRANSMITER (i2c read mode)
case 0xB0:
case 0xA8:  // otrzymano adres, zwrócono ACK,                         ==>> wysyłanie danych
case 0xB8:  // przesłano bajt z TWDR, otrzymano ACK                   ==>> wysyłanie danych (
kolejny bajt)
 mode = twi_cmd & 0xF0;
 switch (mode) {
  case 0x00:
   if (twi_cmd == 0x00) { // 0x0n - change masks
    // bits 0,1,2:    PINB, PINC, PIND    changed
    // bits 3:        3-state input changed
    // bits 4-7:      ADC changed
    TWDR = status_mask;
    status_mask = 0x00;
   }
   break;
  case 0x40:  // CURRENT PIN STABILITY COUNTER
  case 0x50:  // (number of TIMER1_COMPA cycles remaining to wait)
   mode = twi_cmd & 0x1F;
   if (mode < 24)
    TWDR = pin_stability[mode];
   break;
  case 0x60:  // INITIAL PIN STABILITY
  case 0x70:  // (number of TIMER1_COMPA cycles to wait before considering the value of a 
  stable)
   mode = twi_cmd & 0x1F;
   if (mode < 24) {
    TWDR = pin_stability_init[mode];
   } else if (mode == 24) {
    // number of timer ticks in one TIMER1_COMPA cycle
    TWDR = OCR1A;
   } else if (mode == 25) {
    // de-bouncing status
    mode = TCCR1B & 0x07;
    if (mode) {
     TWDR = mode | 0xF0;
    } else {
     TWDR = 0;
    }
   } else if (mode == 26) {
    // input port read / de-bouncing bit-mask
    TWDR = status_input_mask;
   }
   break;
  case 0x80:  // TRI-STATE limits
   mode = twi_cmd & 0x0F;
   if (mode < 4)
    TWDR = adc_normal_min[mode];
   else if (mode < 8)
    TWDR = adc_normal_max[mode - 4];
   else if (mode < 12)
    TWDR = adc_active_min[mode - 8];
   else if (mode < 16)
    TWDR = adc_active_max[mode - 12];
   break;
  case 0x90:
   mode = twi_cmd & 0x0F;
```

```
    switch (mode) {
     // mode & 0x0c == 0b00   =>   PORTx (port x output value / port x pullup resistor mask)
     case 0x01:
      TWDR = PORTB;
      break;
     case 0x02:
      TWDR = PORTC;
      break;
     case 0x03:
      TWDR = PORTD;
      break;
     // mode & 0x0c == 0b01   =>   DDRx (port x direction)
     case 0x05:
      TWDR = DDRB;
      break;
     case 0x06:
      TWDR = DDRC;
      break;
     case 0x07:
      TWDR = DDRD;
      break;
     // mode & 0x0c == 0b10   =>   PINx (port x input value)
     case 0x09:
      TWDR = PINB;
      break;
     case 0x0a:
      TWDR = PINC;
      break;
     case 0x0b:
      TWDR = PIND;
      break;
    }
    break;
   case 0xA0: // ADC
    if (twi_cmd == 0xAF)
     TWDR = status_adc_mode;
    else
     TWDR = status_adc_val[twi_cmd & 0x03];
    break;
   case 0xB0: // PORT B
    TWDR = read_status(status_b,    twi_cmd & 0x0f);
    break;
   case 0xC0: // PORT C
    TWDR = read_status(status_c,    twi_cmd & 0x0f);
    break;
   case 0xD0: // PORT D
    TWDR = read_status(status_d,    twi_cmd & 0x0f);
    break;
   case 0xE0: // TRI-STATE
    TWDR = read_status(status_tri,  twi_cmd & 0x0f);
    break;
  }
  twi_cmd = 0x00;
  ack = 0;
  if (PORTB & _BV( PB4 ))   PORTB &= ~( _BV( PB4 ) );  else   PORTB |= ( _BV( PB4 ) );
  break;
 case 0xC0: // przesłano bajt z TWDR, otrzymano NOT ACK         ==>> oczekiwanie na adres
  break;
 case 0xC8: // przesłano ostatni bajt z TWDR, otrzymano ACK    ==>> oczekiwanie na adres
  break;
```

```
    case 0x00:  // Bus error
      TWCR = (1<<TWSTO)|(1<<TWINT);
      break;

  }

  // clear interrupt flag
  TWCR = (1<<TWEN) | (1<<TWIE) | (1<<TWINT) | (ack<<TWEA);
}

/*
#define OUTPUT_STATUS_SIZE 1
#define INPUT_STATUS_SIZE 3
volatile uint8_t output_status[OUTPUT_STATUS_SIZE];
volatile uint8_t input_status[INPUT_STATUS_SIZE];
volatile uint8_t buf_pos;

ISR( TWI_vect ) {
 uint8_t status = TWSR & 0xf8;
 uint8_t ack = 1;

 switch (status) {
   // RECIVER (i2c write mode)
   case 0x68:
   case 0x60: // otrzymano adres, zwrócono ACK,                 ==>> odbiór danych
     buf_pos = 0;
     break;
   case 0x80: // otrzymano dane, zwrócono ACK                    ==>> odbiór danych (kolejny
   bajt)
     output_status[buf_pos++] = TWDR;
     if (buf_pos >= OUTPUT_STATUS_SIZE) //                        ==>> kolejne dane błędem
       ack = 0;
     break;
   case 0x88: // otrzymano dane, zwrócono NOT ACK                ==>> oczekiwanie na adres
     break;
   case 0xA0: // otrzymano STOP lub powtórzony START             ==>> oczekiwanie na adres
     break;

   // TRANSMITER (i2c read mode)
   case 0xB0:
   case 0xA8: // otrzymano adres, zwrócono ACK,                  ==>> wysyłanie danych
     buf_pos = 0;
   case 0xB8: // przesłano bajt z TWDR, otrzymano ACK          ==>> wysyłanie danych (
   kolejny bajt)
     TWDR = input_status[buf_pos++];
     if (buf_pos >= INPUT_STATUS_SIZE) //                         ==>> zakończenie wysyłania (
     ostatni bajt)
       ack = 0;
     break;
   case 0xC0: // przesłano bajt z TWDR, otrzymano NOT ACK      ==>> oczekiwanie na adres
     break;
   case 0xC8: // przesłano ostatni bajt z TWDR, otrzymano ACK  ==>> oczekiwanie na adres
     break;

   case 0x00: // Bus error
     TWCR = (1<<TWSTO)|(1<<TWINT);
     break;

 }
```

```
 // clear interrupt flag
 TWCR = (1<<TWEN) | (1<<TWIE) | (1<<TWINT) | (ack<<TWEA);
}
*/
```

RUN.SH

```
#!/bin/bash

# do skryptów startowych dodać screen -dm bash –rcfile /bms/run.sh

modprobe i2c-bcm2708
modprobe i2c-dev

# i2cdetect -l
BUS_ADDR="bcm2708_i2c.1"

# i2cdetect $BUS_ADDR

cd /bms/

set -o ignoreeof
alias exit="echo NO EXIT FROM THIS SHELL"
alias logout="echo NO EXIT FROM THIS SHELL"


###
### REJESTRY UKŁADU MCP23008
###
IODIR=0x00;    # kierunek transferu
GPINTEN=0x02;  # generowanie przerwan
DEFVAL=0x03;   # ”normalny” stan wejsc, inna wartosc generuje przerwanie (gy odp. ust INTCON)
INTCON=0x04;   # porownywanie z DEFVAL zamiast stanem poprzednim
IOCON=0x05;    # konfiguracja (tryb pracy wyjścia przerwaniowego, ...)
GPPU=0x06;     # rezystory podciagajace

INTF=0x07;     # piny na ktorych wystapila sytuacja wywolujaca przerwanie
INTCAP=0x08;   # stan GPIO w chwili pierwszego przerwania
GPIO=0x09;     # aktualny stan wejsc i wyjsc


# OBSŁUGA PRZERWANIOWA MCP23008
## konfiguracja - kierunek transferu, generowanie przerwań, rezystory podciągające
## starsze 4 bity jako wejście z przerwaniami, młodsze jako wyjście
#    i2cset -y $BUS_ADDR $DEV_ADDR $IODIR 0xf0
#    i2cset -y $BUS_ADDR $DEV_ADDR $GPINTEN 0xf0
#    i2cset -y $BUS_ADDR $DEV_ADDR $GPPU    0xf0

# ustawienie wartości wyjść
#    i2cset -y $BUS_ADDR $DEV_ADDR $GPIO 0x05

# odczyt wejść
#    i2cget -y $BUS_ADDR $DEV_ADDR $GPIO

# próba obsługi przerwań
#    i2cget -y $BUS_ADDR $DEV_ADDR $INTF; i2cget -y $BUS_ADDR $DEV_ADDR $INTCAP; i2cget -y ↩
$BUS_ADDR $DEV_ADDR $GPIO
```

```
# Niestety wbrew dokumentacji układ w INTF nie odnotowuje kolejnych pinów,
# które wywołują kolejne przerwanie (w trakcie trwania przerwania).
# Jest tam odnotowany tylko pierwszy pin który wywołał przerwanie
# (po wyczyszczeniu przerwań odczytem GPIO lub INTCAP).
#
# Jest to znaczna niedogodność w zastosowaniach wymagających szybkiego
# wychwytywania krótkich przerwań przy wolnej obsłudze I2C.
# Gdyby układ działał zgodnie z dokumentacją mugłbby robić za bufor przerwań
# dla układów nie mających podtrzymywania wystawionego przerwania,
# ale wymagających przekazania informacji o nim (np. wejścia linii alarmowych)


###
### KONFIGURACJA MODUŁÓW DO STEROWANIA PRZEKAŹNIKAMI
###
for DEV_ADDR in 0x24 0x25 0x26 0x27; do
 # wyjscia w stanie niskim
 i2cset -y $BUS_ADDR $DEV_ADDR $GPIO  0x00
 # output
 i2cset -y $BUS_ADDR $DEV_ADDR $IODIR 0x00
done

# bity związane z kolorami kabli do przekaznikow
przek_get_bit() {
 case $1 in
  "bp") echo 0x80 ;;
  "p")  echo 0x40 ;;
  "bn") echo 0x20 ;;
  "n")  echo 0x10 ;;
  "bz") echo 0x08 ;;
  "z")  echo 0x04 ;;
  "bb") echo 0x02 ;;
  "b")  echo 0x01 ;;
 esac
}

# adresy i2c związane z modułami sterowania przekaźnikami
przek_get_addr() {
 case $1 in
  1) echo 0x25 ;;
  2) echo 0x24 ;;
  3) echo 0x26 ;;
  4) echo 0x27 ;;
 esac
}

# maski bitowe przechowujące stan przekaźników i funkcje na nich operujące
[ "${przek_bitmask[*]}" = "" ] && przek_bitmask=(0 0 0 0 0 0 0 0)

# funkcja ustawiająca stan przekaźnika
przek2() {
 addr=`przek_get_addr $1`
bit=$2
 if [ $3 -eq 1 ]; then
  przek_bitmask[$1]=$(( ${przek_bitmask[$1]} | $bit ))
 else
  przek_bitmask[$1]=$(( ${przek_bitmask[$1]} & ~ $bit ))
 fi
 i2cset -y $BUS_ADDR $addr $GPIO  ${przek_bitmask[$1]}
```

```
}
przek() {
 bit=`przek_get_bit $2`
przek2 "$1" "$bit" "$3"
}


###
### KONFIGURACJA MODUŁU 1WIRE
###
DEV_ADDR=0x20
i2cset -y $BUS_ADDR $DEV_ADDR $IODIR 0xa8 # 0-2: OUT (wybor linii), 4,6: OUT, 3,5,7: IN
i2cset -y $BUS_ADDR $DEV_ADDR $GPIO  0x00
i2cset -y $BUS_ADDR $DEV_ADDR $GPPU  0xa8 # all input with pull-up

onewire_line=0
set_onewire() {
 if [ $1 -ge 0 -a $1 -lt 8 ]; then
  onewire_line=$1
  i2cset -y $BUS_ADDR 0x20 $GPIO $(( $onewire_line | $sygn_orange_line ))
 fi
}


###
### KONFIGURACJA MODUŁÓW ATMEGA
###
for DEV_ADDR in 0x10 0x11; do
 # port D as output (DDRD=0xff)
 i2cset -y $BUS_ADDR $DEV_ADDR 0x97 0xff
 # wyjscia w stanie niskim
 i2cset -y $BUS_ADDR $DEV_ADDR 0x93 0x00

# port C jako input z pull-up
 i2cset -y $BUS_ADDR $DEV_ADDR 0x96 0x00
 i2cset -y $BUS_ADDR $DEV_ADDR 0x92 0xff
done
 # port B jako input z pull-up
 i2cset -y $BUS_ADDR 0x10 0x95 0x40
 i2cset -y $BUS_ADDR 0x10 0x91 0xbf
 i2cset -y $BUS_ADDR 0x11 0x95 0x00
 i2cset -y $BUS_ADDR 0x11 0x91 0x3e


# maska bitowa przechowujące stan LEDów
led_val=0

# funkcje ustawiające stan LEDów oraz realizujące miganie
led2() {
 if [ $2 -eq 1 ]; then
  led_val=$(( $led_val | $1 ))
 else
  led_val=$(( $led_val & ~ $1 ))
 fi
 i2cset -y $BUS_ADDR 0x11 0x93 $led_val
}
led() {
 bit=`led_get_bit $1`
led2 $bit $2
}
```

```
led_blink() {
 # usage: led_blink led_bits time count
 led_val_on=$(( $led_val | $1 ))
 led_val_off=$(( $led_val & ~ $1 ))

for i in `seq 1 $3`; do
  i2cset -y $BUS_ADDR 0x11 0x93 $led_val_on
  sleep $2
  i2cset -y $BUS_ADDR 0x11 0x93 $led_val_off
  [ $i -eq $3 ] || sleep $2
 done
}

### ODCZYT DI

# inicjalizacja na    stany normalne (bez wcisnietych przyciskow itp)
DI_A=0x28; DI_B=0xba; DI_C=0x3e;

# obliczmy maske stanu normalnego
DI_normal_mask=$(( ($DI_A & 0xa8) << 13 | $DI_B << 8 | $DI_C ))
DI_bistable_mask=$(( 0x2 | 0x4 | 0x40000 | 0x100000 ))

DI_status=0; DI_cnt=0; DI_bistable_diff=0; DI_bistable_cnt=0;
DI_read() {
 # zapamietujemy poprzedni status wejść jako DI_status_old
 DI_status_old=$DI_status

 # odczytujemy wszystkiewejscia cyfrowe (gdy błąd odczytu to uznajemy że brak zmiany)
 t=`i2cget -y $BUS_ADDR 0x20 $GPIO 2>/dev/null` && DI_A=$t
 t=`i2cget -y $BUS_ADDR 0x10 0x99  2>/dev/null` && DI_B=$t
 t=`i2cget -y $BUS_ADDR 0x11 0x99  2>/dev/null` && DI_C=$t

 # wyliczamy status oparty na wszystkich wejściach (w normalnym stanie = 0)
 t=$(( ($DI_A & 0xa8) << 13 | ($DI_B & 0xbf) << 8 | $DI_C ))
 DI_status=$(( $t ^ $DI_normal_mask ))

 # obliczamy zmianę
 DI_diff=$(( $DI_status ^ $DI_status_old ))


 if [ $DI_bistable_cnt -eq 1 -a $DI_bistable_diff -ne 0 -a \
      $(( ($DI_status ^ $DI_status_old) & $DI_bistable_mask )) -eq 0 ]; then
  # jeżeli w poprzednim cyklu była zmiana na bistabilnych i się utrzymała to zaznaczamy
  do realizacji
  DI_bistable_cnt=2
 else
  # jak nie to sprawdzamy czy w obecnym cyklu jest zmiana na bistabilnych
  DI_bistable_diff=$(( $DI_diff & $DI_bistable_mask ))

  if [ $DI_bistable_diff -ne 0 ]; then
   DI_bistable_cnt=1
  else
   DI_bistable_cnt=0
  fi
 fi


 t=$(( $DI_status & ~ $DI_bistable_mask ))
 if [ $t -ne 0 ]; then
  # jeżeli jest wciśnięty któryś z monostabilnych
```

```
  if [ $DI_check -eq 0 ]; then
   # jeżeli pierwsza rejestracja wciśnięcia - zapamiętaj który to przycisk
   DI_check=$t
  fi

  # jeżeli ten który zapamiętaliśmy pozostaje wciśnięty to zwiększ licznik
  if [ $(( $t & $DI_check )) -ne 0 ]; then
   let DI_cnt++;
  fi
 else
  # gdy nie ma wciśniętego monostabilnego wyzeruj liczniki i zapamiętany przycisk
  DI_cnt_old=$DI_cnt
  DI_cnt=0
  DI_check=0
 fi
}

DI_read_loop() {
 while true; do
  # odczyt linii DI
  DI_read

  if [ $DI_bistable_cnt -eq 2 ]; then
   # wykryto zmianę stanu przycisku bistabilnego
   led_blink 0xc8 0.15 1
   DI_action0 $DI_bistable_diff $DI_status_old
  elif [ $DI_cnt -eq 0 ]; then
   # wykryto zmianę stanu przycisku monostabilnego
   if [ $DI_cnt_old -ge 20 ]; then
    # bardzo długie naciśnięcie
    DI_action4 $DI_status_old
   elif [ $DI_cnt_old -ge 14 ]; then
    # długie naciśnięcie
    DI_action3 $DI_status_old
   elif [ $DI_cnt_old -ge 8 ]; then
    # średnie naciśnięcie
    DI_action2 $DI_status_old
   elif [ $DI_cnt_old -ge 2 ]; then
    # krótkie nasićnięcie (potwierdzone dwoma odczytami)
    DI_action1 $DI_status_old
   fi
  elif [ $DI_cnt -eq 2 ]; then
   # wykryto nacisniecie przycisku
   led_blink 0xc8 0.15 1
  elif [ $DI_cnt -eq 8 ]; then
   # naciśnięcie przycisku trwa ~1s
   led_blink 0xc8 0.15 2
  elif [ $DI_cnt -eq 14 ]; then
   # naciśnięcie przycisku trwa ~2s
   led_blink 0xc8 0.15 3
  elif [ $DI_cnt -eq 20 ]; then
   # naciśnięcie przycisku trwa ~3s
   led_blink 0xc8 0.15 4
  elif [ $DI_cnt -eq 40 ]; then
   # naciśnięcie przycisku trwa zbyt długo
   DI_action_verbose $DI_status_old
   led_blink 0xc8 0.4 5
   DI_cnt=0
   DI_cnt_old=0
  fi
```

```
   sleep 0.05
  done
}


###
### ZAINCLUDOWANIE FUNKCJI REALIZUJĄCYCH STEROWANIE
###
. ./output.sh
. ./input.sh

# don’t run in background - we want have variables used by DI_read_loop system in current ⤸
shell
# eg for ‘set | egrep ’^output_state|^przek_bitmask’‘
DI_read_loop
```

OUTPUT.SH

```
#!/bin/bash

[ "${output_state[*]}" = "" ] && declare -A output_state=()
output() {
 if [ $2 == "switch" ]; then
  if [ ${output_state[$1]:-0} -eq 0 ]; then
   action=1
  else
   action=0
  fi
 else
  action=$2
 fi
 output_state[$1]=$action

case $1 in
  "pokoj1")      ./modbus_rtu/write_USB2 12 19 $2 ;;
  "pokoj2")      przek 1 bn $2 ;;
  *)
   echo "USAGE: $0 pokoj1 0|1|switch|val"
   echo "USAGE: $0 pokoj2 0|1|switch"
   ;;
 esac
}

roleta() {
 case $1 in
  "up")
   przek 4 bp 1
   przek 4 p  1
   ;;
  "down")
   przek 4 bp 0
   przek 4 p  1
   ;;
  "stop")
   przek 4 bp 0
   przek 4 p  0
   ;;
  *)
```

```
   echo "USAGE: $0 up|down|stop"
   ;;
 esac
}


# ledy w panelach z przyciskami
led_get_bit() {
 case $1 in
  "panel1")                echo 0x40 ;;
  "panel2")                echo 0x80 ;;
 esac
}

# sygnalizator podłączony do GPIO na module 1wire
sygn_orange_line=0
sygn_orange() {
 case $1 in
  "on")
   sygn_orange_line=0x40
   ;;
  "slow")
   sygn_orange_line=0x10
   ;;
  "off")
   sygn_orange_line=0x00
   ;;
 esac
 i2cset -y $BUS_ADDR 0x20 $GPIO $(( $onewire_line | $sygn_orange_line ))
}
```

INPUT.SH

```
#!/bin/bash

DI_action0() {
 printf "X: 0x%x 0x%x\n" $1 $2
 if [ $(( $1 & 0x4)) -ne 0 ]; then  # zmiana była na 0x4
  if [ $(( $2 & 0x4)) -ne 0 ]; then #    został aktywowany
   if [ $(( $2 & 0x2)) -ne 0 ]; then
    roleta up
   else
    roleta down
   fi
  else                                #    został dezaktywowany
   roleta stop
  fi
 fi
}

DI_action1() {
 printf "A: 0x%x\n" $1
 [ $(( $1 & 0x120)) -ne 0 ]    && swiatlo "pokoj1" switch
[ $(( $1 & 0x1000)) -ne 0 ]    && gniazdko "komputer" 1
}
DI_action2() {
 printf "B: 0x%x\n" $1
 [ $(( $1 & 0x20)) -ne 0 ]     && swiatlo "pokoj2" switch
```

```
[ $(( $1 & 0x1000)) -ne 0 ]   && echo "unused"
}
DI_action3() {
 printf "C: 0x%x\n" $1
 [ $(( $1 & 0x1000)) -ne 0 ]   && gniazdko "komputer" 0
}
DI_action4() {
 printf "D: 0x%x\n" $1
}
DI_action_verbose() {
 printf "INPUT: 0x%x\n" $1
 [ $(( $1 & 0x1)) -ne 0 ]      && echo " przycisk 1 modul Atmega"
[ $(( $1 & 0x200000)) -ne 0 ] && echo " przycisk 1 modul 1wire"
[ $(( $1 & 0x800000)) -ne 0 ] && echo " przycisk 2 modul 1wire"
}


verbose_read_tmperature() {
 function t() {
  echo -n "$1"
  shift 1
  ./one_wire/read_temp $@ | grep TEMP | cut -f2- -d:
 }
 # kabel nr 5 - temperatura w racku
 set_onewire 0
 t "rack dolny front:            " 0x28 0x34 0x8B 0x58 0x02 0x00 0x00 0x1F
 t "rack dolny tył:              " 0x28 0xFA 0x52 0x58 0x02 0x00 0x00 0x3E
 set_onewire 3
 t "rack górny tył:              " 0x28 0x8A 0x39 0x72 0x02 0x00 0x00 0xAC
 t "rack górny front:            " 0x28 0xC7 0xB3 0x71 0x02 0x00 0x00 0x6D
}

verbose_read_analog() {
 DEV_ADDR=0x10
 i2cset -y $BUS_ADDR $DEV_ADDR 0xAf 0xf3;
 echo -n "POTENCJOMETR: "; i2cget -y $BUS_ADDR $DEV_ADDR 0xA0;
 echo -n "PIR 1 "; i2cget -y $BUS_ADDR $DEV_ADDR 0xA1;
 echo -n "DYM 1  "; i2cget -y $BUS_ADDR $DEV_ADDR 0xA2;
}


verbose_read_input() {
 if invalA=`i2cget -y $BUS_ADDR 0x20 $GPIO`; then
  [ $(( $invalA & 0x20)) -eq 0 ] && echo "przycisk 1 modul 1wire"
  [ $(( $invalA & 0x80)) -ne 0 ] && echo "przycisk 2 modul 1wire"
else
  echo "ERROR: A"
fi
 if invalB=`i2cget -y $BUS_ADDR 0x10 0x99`; then
  [ $(( $invalB & 0x01)) -ne 0 ] && echo "przycisk 1 modul Atmega"
else
  echo "ERROR: B"
fi
}
```

Manipulator do systemu alarmowego

Jest to projekt manipulatora dla systemu alarmowego. Manipulator wyposażony jest w klawiaturę 4x4, wyświetlacz LCD, dwie diody sygnalizacyjne, buzer. Komunikacja z jednostką centralną odbywa poprzez RS485, z wykorzystaniem protokołu MODBUS RTU. Manipulator wyłącznie przesyła dane o wprowadzonym kodzie do centralki oraz reaguje na polecenia otrzymywane od niej (w szczególności dotyczące sterowania wyświetlaczem LCD). Konstrukcja szyfratora umożliwia montaż klawiatury po jednej stronie ściany a płytki z mikrokontrolerem po drugiej. Rozwiązanie takie stanowi dodatkowe zabezpieczenie manipulatora przed próbami niepowołanego dostępu.

- schemat ideowy[2002](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/manipulator_alarmowy/manipulator.pdf`) oraz projekt płytki[2003](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/manipulator_alarmowy/manipulator.pcb.pdf`) manipulatora
- schemat ideowy układu klawiatury[2004](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/manipulator_alarmowy/manipulator-klawiatura.pdf`)
- program sterujący praca manipulatora
- projekt wymaga fragmentów kodu oraz modułu elektronicznego z Atmega8 z biblioteczki dla uC

Pobierz[2005]

MANIPULATOR.C

```
/*
 * program sterownika manipulator (szyfratora) systemu alarmowego
 * z komunikacją poprzez port RS oraz obsługą wyświetlacza LCD
 * dla mikrokontrolera Atmega8
 *
 * wymaga plików odpowiedzialnych za obsługę komunikacji modbus
 * (z użyciem FreeMODBUS) i budowę całości projektu z:
 * http://opcode.eu.org/usage_and_config/toolbox4electronic/libs4avr/
 *
 * kompilacja i wgranie przez ISP z użyciem USBasp:
 *    make sterownik_oswietlenia flash
 *
 * konfiguracja układu na zewnętrzny kwarc 8MHz (lub troche mniejszy)
 *    sudo avrdude -c usbasp -p m8 -U lfuse:w:0xef:m
 *
 */

#include <inttypes.h>
#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

/** nagłówki obsługi MODBUS RTU **/
#include "modbus.c"

#define VERSION        0x0002
#define SLAVE_ADDR     0x0A
#define SLAVE_BASE_REG 0x0010


/** nagłówki obsługi wyświetlacza LCD **/
#define lcd_send(byte) PORTC = byte
#define LCD_E         0x02
#define LCD_RS_DATA   0x01
#define LCD_RS_CMD    0x00
#define LCD_DATA_OFFSET 2
#include "lcd.c"
```

```
#define LINE_LEN      16
#define STATUS_LEN    3*LINE_LEN+2

void short_beep(int8_t wait1, int8_t wait2) {
 int8_t i;
 PORTD |= 1 << PD3;
 for (i=0; i<wait1; i++) _delay_ms(30);
 PORTD &= ~(1 << PD3);
 for (i=0; i<wait2; i++) _delay_ms(30);
}


volatile uint8_t  status[ STATUS_LEN + 1 ];
/** status[0 ... LINE_LEN-1]                    = data from keyboard
    status[LINE_LEN]                            = mode
                                                    0      : reserved for internal use (initialising ⤸
                                                    state)
                                                  1 -   99 : show message
                                                100 - 149 : show message and wait for enter number
                                                150 - 199 : show message and wait for enter secret
                                                200 - 250 : action (state don't storage)
                                                   200     : backlight pernament ON
                                                   201     : backlight timer ON
                                                   203     : backlight OFF
                                                   204     : beeper ON
                                                   205     : beeper - short beep          short_beep(4, 0)⤸
                                                   ;
                                                   206     : beeper - two short beep      short_beep(4, 6)⤸
                                                   ; short_beep(4, 0);
                                                   207     : beeper - long beep           short_beep(7, 0)⤸
                                                   ;
                                                   208     : beeper - extra long beep     short_beep(20, ⤸
                                                   0);
                                                   209     : beeper OFF
                                                   210     : led 1 ON
                                                   211     : led 1 OFF
                                                   212     : led 2 ON
                                                   213     : led 2 OFF
                                                   214     : led 3 ON
                                                   215     : led 3 OFF
                                                   216     : clear LCD message buffer (status[LINE_LEN+⤸
                                                   2    ... 3*LINE_LEN+1])
    status[LINE_LEN+1]                          = number of valid chars in data from keyboard
    status[LINE_LEN+2      ... 2*LINE_LEN+1]    = message first line
    status[2*LINE_LEN+2 ... 3*LINE_LEN+1]       = message second line
**/


volatile uint8_t  new_mode = 0;


void reset_display( void ) {
 lcd_clear();
 _delay_ms(25);

switch (status[LINE_LEN]) {
  case 0:
   lcd_send_string("Initialising ...", 16);
   break;
  case 1:
```

```
    lcd_send_string("Invalid PIN", 11);
    break;
   case 100:
    lcd_send_string("enter CMD CODE", 14);
    break;
   case 150:
    lcd_send_string("enter PIN", 9);
    break;
   case 99:
   case 149:
   case 199:
    lcd_send_string((char *)((uint8_t*)status + LINE_LEN+2), LINE_LEN);
    break;
 }

lcd_second_line();

if(status[LINE_LEN] == 99)
  lcd_send_string((char *)((uint8_t*)status + 2*LINE_LEN+2), LINE_LEN);
}

void set_new_mode( void ) {
 int i;

if (new_mode < 200) {
  status[LINE_LEN]   = new_mode;
  status[LINE_LEN+1] = 0;
  reset_display();
 } else {
  switch (new_mode) {
   case 200:
    PORTD |= 0x04;
    TIMSK &= ~_BV( OCIE1A );
    break;
   case 201:
    PORTD |= 0x04;
    TIMSK |= _BV( OCIE1A );
    TCNT1 = 0x0000;
    break;
   case 202:
    PORTD &= 0xfb;
    TIMSK |= _BV( OCIE1A );
    TCNT1 = 0x0000;
    break;

   case 204:
    PORTD |= 1 << PD3;
    break;
   case 206:
    short_beep(4, 6);
   case 205:
    short_beep(4, 0);
    break;
   case 207:
    short_beep(7, 0);
    break;
   case 208:
    short_beep(20, 0);
    break;
   case 209:
```

```
    PORTD &= ~(1 << PD3);
    break;

   case 210:
    PORTB |= ( _BV( PB3 ) );
    break;
   case 211:
    PORTB &= ~( _BV( PB3 ) );
    break;
   case 212:
    PORTB |= ( _BV( PB4 ) );
    break;
   case 213:
    PORTB &= ~( _BV( PB4 ) );
    break;
   case 214:
    PORTB |= ( _BV( PB5 ) );
    break;
   case 215:
    PORTB &= ~( _BV( PB5 ) );
    break;

   case 216:
    for (i=LINE_LEN+2; i<3*LINE_LEN+2; i++)
     status[i] = ' ';
  }
 }
 new_mode = 0;
}

int main( void ) {
 uint8_t i = 0;

// port B -output - klawiatura, sterowanie RS, diody
 DDRB = 0xff;
 PORTB = 0;

// port C - output - sterowanie LCD
 DDRC = 0xff;
 PORTC = 0;

// port D - output (pin 2,3) - buzzer, podświetlenie, input (pin4-7) - klawiatura
 DDRD = 0x0f;
 PORTD = 0xfc;

// initial wait and "BEEP"
 for (i=0; i<33; i++) _delay_ms(30);
 PORTD = 0xf4;

// initialise status
 for (i=0; i<STATUS_LEN; i++)
  status[i]=0;

// initialise LCD display
 lcd_set_4bit_2line();
 // wait and re-initialise LCD display ... i don't know why but this LCD need this ...
 _delay_ms(30); lcd_set_4bit_2line();
 lcd_set_display_on_cursor_off();
 lcd_set_move_right();
 reset_display();
```

```
// counter - 15k ticks per second == 150 ticks per pwm period    (for F_CPU ~= 3MHz)
  TCNT1 = 0x0000;
 TIMSK |= _BV( OCIE1A ); // enable interrupt from this timer
 OCR1A = 14648; // 5s
 TCCR1B = _BV( CS10 ) | _BV( CS12 ); // timer ticks = F_CPU / 1024

// modbus
 eMBInit( MB_RTU, SLAVE_ADDR, 0, 9600, MB_PAR_NONE );
 sei();
 eMBEnable(  );

// initializion finish "BEEP"
 short_beep(3, 0);

new_mode = 100;

uint8_t row = 0, col = 0, kbd = 0, last_kbd = 0;
 uint8_t kbd_cnt[4] = {0, 0, 0, 0};
 while(1) {
  if (new_mode) {
   set_new_mode();
  }

  if (++i > 250) {
   i = 0;

   // podsumowujemy głosowanie
   kbd = 255;
   for (row=0; row<4; row++) {
    // gdy 75% trafień w iedną wartość to uznaiemy ią za poprawną
    if (kbd_cnt[row] > 0xbd) {
     if (col != 3) {
      if (row != 3) {
       kbd = '0' + col + 1 + row * 3;
      } else { // last row:   *, 0, #
       if (col == 0)
        kbd = '*';
       else if (col == 2)
        kbd = '#';
       else
        kbd = '0';
      }
     } else { // last col:   A, B, C, D
      kbd = 'A' + row;
     }
    }
    kbd_cnt[row] = 0;
   }

   // jeżeli głosowanie wyłoniło nowy klawisz
   if (kbd != 255 && kbd != last_kbd) {
    // start backlight
    PORTD |= 0x04;
    TCNT1 = 0x0000;
    if (status[LINE_LEN] > 99 ) {
     // when entering new key ...
     if (kbd == '*') {
      // clear
      status[LINE_LEN+1] = 0;
```

```
      short_beep(7, 0);
      reset_display();
     } else {
      if (status[LINE_LEN+1] < LINE_LEN) {
       // save key-code to status register
       status[ status[LINE_LEN+1]++ ] = kbd;

       if (status[LINE_LEN] < 150 ) {
        lcd_send_data(kbd);
       } else {
        lcd_send_data('*');
       }
       short_beep(3, 0);
      }
     }
    }
   }

   // zapamiętujemy bierzący klawisz (lub jego brak)
   last_kbd = kbd;

   // przechodzimy do następnej kolumny
   if (++col > 3)
    col = 0;

   // na wybrany wiersz podawana jest masa
   PORTB = (PORTB & 0xfc) | col;
  } else {
   // odczytujemy wejście i liczymy głosy
   kbd = PIND & 0xf0;
   switch (kbd) {
    case 0xe0:
     kbd_cnt[0]++;
     break;
    case 0xd0:
     kbd_cnt[1]++;
     break;
    case 0xb0:
     kbd_cnt[2]++;
     break;
    case 0x70:
     kbd_cnt[3]++;
     break;
   }
  }

  eMBPoll();
 }
}

// timer
ISR(TIMER1_COMPA_vect, ISR_NOBLOCK) {
 PORTD &= 0xfb; // disable lcd backlight
}

// modbus read/write callback
eMBErrorCode eMBRegHoldingCB( UCHAR * pucRegBuffer, USHORT usAddress, USHORT usNRegs, ⤸
eMBRegisterMode eMode ) {
 // zapytanie dotyczy usNRegs rejestrów począwszy od rejestru usAddress
 uint8_t line = usAddress - SLAVE_BASE_REG - 1;
```

```
 uint8_t eline = line + usNRegs;

if (eMode == MB_REG_READ) {
  if (line >=0 && eline < STATUS_LEN) {
   for (; line< eline; line++) {
    *pucRegBuffer++ = status[2*line];
    *pucRegBuffer++ = status[2*line+1];
   }
   return MB_ENOERR;
  } else if (usAddress == 0xf0ff) {
   *pucRegBuffer++ = VERSION >> 8;
   *pucRegBuffer++ = VERSION & 0xff;
   return MB_ENOERR;
  }
 } else if (eMode == MB_REG_WRITE) {
  if (2*line == LINE_LEN) { // set mode
   new_mode = *(pucRegBuffer+1);
   return MB_ENOERR;
  } else if (line >LINE_LEN/2 && eline < STATUS_LEN/2) { // set data for LCD
   status[2*line]   = *pucRegBuffer++;
   status[2*line+1] = *pucRegBuffer;
   return MB_ENOERR;
  }
 }
 return MB_ENOREG;
}
```

Triakowy sterownik obwodów 230V AC

Sterownik umożliwia sterowanie wieloma obwodami 230V AC poprzez interfejs RS-485[2006] z protokołem modbus-rtu, z wykorzystaniem mikrokontrolera Atmega8 i zestawy optotriak + triak. W zależności od wariantu realizacyjnego może to być od 6 linii PWM (z fazową regulacją jasności w technologii Pulse Width Modulation[2007]), poprzez 4 PWM + 4 ON/OFF[2008](PDF: http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/sterownik_oswietlenia-4PWM.pdf), do 12 linii ON/OFF. Dla modułów triakowych przewidziano zastosowanie BTA16 (lub BTB16) pracujących na 10A z radiatorem HS-142-50 (patrz artykuł o doborze radiatorów).

Projekt tworzony był z myślą o sterowaniu obwodów oświetleniowych jednak z powodzeniem może być zastosowany do sterowania innymi obwodami AC, zwłaszcza o niewielkich prądach (gdzie straty mocy na triaku nie są wada w porównaniu z rozwiązaniami przekaźnikowymi). Wchodzące w skład projektu moduły triakowe mogą być też wykorzystane jako autonomiczne moduły załączające (umieszczone np. w standardowych obudowach na szynę DIN) i sterowane bezpośrednio z innego sterownika (np. opartego na procesorze ARM i systemie Linux).

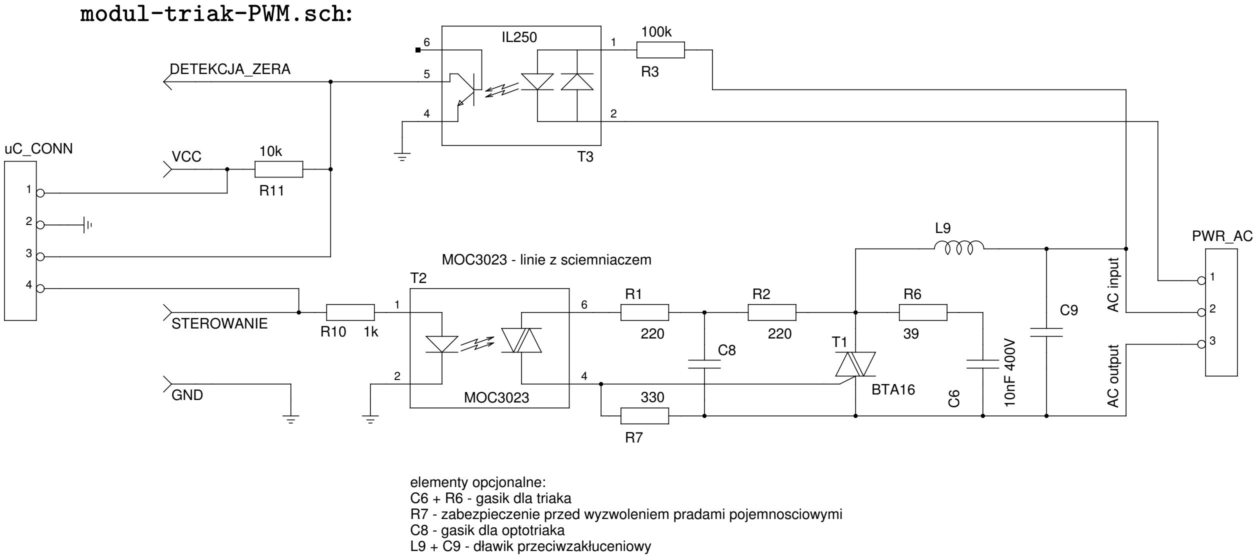
Pobierz[2009]

Na projekt sterownika składają się moduły:

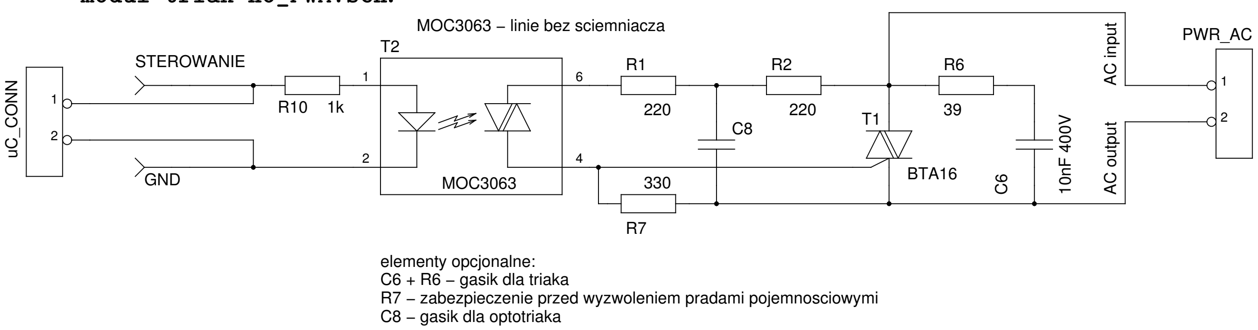
- triak + optotriak + układ detekcji zera dla linii PWM:
  - schemat ideowy[2010](PDF: http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-PWM.pdf)
  - projekt PCB[2011](PDF: http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-PWM.pcb.pdf)
  - Pliki do wykorzystania przy tworzeniu projektu zbiorczego: symbol[2012], schemat[2013] wstawiający złącze do podłączenia modułu na osobnym PCB
- triak + optotriak z wbudowaną detekcją zera dla linii ON/OFF:
  - schemat ideowy[2014](PDF: http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-no_PWM.pdf)
  - projekt PCB[2015](PDF: http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-no_PWM.pcb.pdf)
  - Pliki do wykorzystania przy tworzeniu projektu zbiorczego: symbol[2016], schemat[2017] wstawiający złącze do podłączenia modułu na osobnym PCB

— mikrokontroler Atmega8 + komunikacja poprzez interfejs RS-485[2018] z projektu biblioteczki dla uC obudowane złaczami i peryferiami do podłączenia 4 modułów PWM i 4 modułów ON/OFF:
  — schemat ideowy[2019](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/sterownik_oswietlenia-4PWM.pdf`)
  — projekt PCB[2020](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/sterownik_oswietlenia-4PWM.pcb.pdf`)
— program sterujący pracą mikrokontrolera

`modul-triak-PWM.sch:`



`modul-triak-no_PWM.sch:`



Zobacz w Sieci: System modułowy z komunikacją po RS 4xx[2021].

Dostosowanie do obciążeń indukcyjnych - gasiki RC

Projekty modułów triakowych uwzględniają (opcjonalne) gasiki RC, zrealizowanie których powinno umożliwić stosowanie sterownika z obciążeniem indukcyjnym (testy zostaną przeprowadzone wkrótce). Układ RC (C6, R6 na schemacie wersji PWM[2022](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak--PWM.pdf`) i ON/OFF[2023](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak--no_PWM.pdf`)) włączony szeregowo do triaka pełni rolę ogranicznika przepięć oraz zapewnia poprawność wyłączenia triaka. W projekcie przewidziano także opcjonalne zabezpieczenie przed wyzwoleniem układu prądami pojemnościowymi (R7) oraz gasik optotriaka (C8 wraz z R1).

Przy stosowaniu tego typu układów należy zapewnić rezystor odpowiedniej mocy oraz (aby uniknąć bzyczenia) kondensator na napięcie przemienne (lub odpowiednio duże napięcie). W przypadku gdy prąd pobierany przez gasik jest problemem można rozważyć użycie drugiego triaka lub przekaźnika (połączonego szeregowo i załączanego wcześniej a odłączanego później) nie posiadającego takiego układu.

W projekcie wersji PWM[2024](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak--PWM.pdf`) przewidziano także opcjonalny układ filtra LC (L9, C9), który zabezpiecza przed propagowaniem zakłóceń do sieci energetycznej gdy na skutek regulacji PWM następuje systematyczne załączanie odbiornika znacznej mocy nie w zerze.

Zobacz w sieci: Jak działa układ gasikowy do stycznika?[2025], dwojnik RC do triaka (obciaz. indukcyjne)[2026], Triak, sterowanie fazowe i gasiki RC[2027], AN1048/D - RC Snubber Networks For Thyristor Power Control and Tran-

sient Suppression[2028], Application Note AN-3008 RC Snubber Networks for Thyristor Power Control and Transient Suppression[2029]

STEROWNIK_OSWIETLENIA.C

```
/*
 * program sterownika oświetlenia z funkcją zdalnego sterowania poprzez UART
 * z użyciem protokołu MODBUS RTU dla mikrokontrolera Atmega8
 *
 * wymaga plików odpowiedzialnych za obsługę komunikacji modbus
 * (z użyciem FreeMODBUS) i budowę całości projektu z:
 * http://opcode.eu.org/usage_and_config/toolbox4electronic/libs4avr/
 *
 * kompilacja i wgranie przez ISP z użyciem USBasp:
 *    make sterownik_oswietlenia flash
 *
 * konfiguracja układu na zewnętrzny kwarc 3-8MHz
 *    sudo avrdude -c usbasp -p m8 -U lfuse:w:0xef:m
 */

#include "modbus.c"

#define VERSION        0x0014
#define SLAVE_ADDR     0x0C
#define SLAVE_BASE_REG 0x0010
#define SLAVE_NUM_REG  9

#define ON_DELAY 16

// otput lines status
volatile uint16_t  status[SLAVE_NUM_REG];

// pwm counters
volatile uint8_t   counters[4];

// led counter
volatile uint16_t  led_cnt;


int main( void ) {
 led_cnt = 0;
 counters[0] = 0;
 counters[1] = 0;
 counters[2] = 0;
 counters[3] = 0;

status[0] = 0;
 status[1] = 0;
 status[2] = 0;
 status[3] = 0;

status[4] = 0;
 status[5] = 0;
 status[6] = 0;
 status[7] = 0;

status[8] = 0xff;

DDRB = 0xfc; // PB0, PB1 input
```

```
 PORTB = 0;
 DDRC = 0xcf; // PC4, PC5 input
 PORTC = 0;
 DDRD = 0xf3; // PD2, PD3 input
 PORTD = 0;

/*
 // interrupt from PD2 (int0)
 GIMSK = (1 << INT0);
 MCUCR = (1 << ISC00) | (1 << ISC01);
 // reagujemy na zbocze narastajace, a nie opadajace (1 << ISC01) i czekamy chwile w ⤸
 procedurze obslugi przerwania
 */

// counter - 15k ticks per second == 150 ticks per pwm period    (for F_CPU ~= 3MHz)
 TCNT1 = 0x0000;
 TIMSK |= _BV( OCIE1A ); // enable interrupt from this timer
 OCR1A = 3;
 TCCR1B = _BV( CS10 ) | _BV( CS11 ); // timer ticks = F_CPU / 64

// modbus
 eMBInit( MB_RTU, SLAVE_ADDR, 0, 9600, MB_PAR_NONE );
 sei();
 eMBEnable(  );

while(1) {
  eMBPoll();
 }
}

// PWM counter (step) interrupt   +   softwere detect zero-cross (and reinit PWM)
ISR(TIMER1_COMPA_vect, ISR_NOBLOCK) {
 uint8_t i, zero_cross_status, zero_cross_mask;

PORTB |= ( _BV( PB5 ) ); // for debuging

// programowa detekcja zbocza narastającego na pinach detekcji zera
  // nie używamy sprzętowego gdyż mamy tylko 2 a potrzebujemy 4
  // a ciężko zsumować te sygnały do jednego sygnału podłączonego do int0
  // (problem jest to że w przypadku braku zasilania cały czas mamy wykrywane zero ...)
 zero_cross_status = ((PINB & 0x03) << 2) | ((PINC & 0x30) >> 4);
 zero_cross_mask = (status[8] ^ zero_cross_status) & zero_cross_status;
   // interesuje nas zmiana w wyniku której mamy stan wysoki
 if (zero_cross_mask) {
  PORTB |= ( _BV( PB3 ) ); // for debuging
  for (i=0; i<4; i++) {
   if (zero_cross_mask & (1 << i)) {
    if (status[i] == 1) {
     // ON
     PORTC |= ( 1 << i );
     counters[i] = 0;
    } else if (status[i] == 0) {
     // OFF
     PORTC &= ~( 1 << i );
     counters[i] = 0;
    } else {
     // start/init PWM coundown
     counters[i] = status[i];
    }
   }
```

```
    }
    PORTB &= ~( _BV( PB3 ) ); // for debuging
  }
  status[8] = zero_cross_status;


// obsługa ściemniacza
  for (i=0; i<4; i++) {
    if (counters[i] > 2) {
      counters[i]--;
    } else if (counters[i] == 2) {
      // ON
      PORTC |= ( 1 << i );
      counters[i] = 1;
    } else if (counters[i] == 1) {
      // OFF
      PORTC &= ~( 1 << i );
      counters[i] = 0;
    }
  }

// obsługa diody sygnalizującej pracę
  led_cnt++;
  if (led_cnt == 15625) {
    // LED off
    PORTB &= ~( _BV( PB4 ) );
  } else if (led_cnt == 31250) {
    // LED on
    PORTB |= ( _BV( PB4 ) );
    // restart LED counter
    led_cnt = 0;
  }

// reste timera
  TCNT1 = 0x0000;

PORTB &= ~( _BV( PB5 ) ); // for debuging

/*
    *   - sygnał z IL250 trwa 1.7ms - 1.88ms
    *   - stabilne załączenie triakaokoło 2.38ms (niestabilne ~2.14ms)
    *   - detekcja zera po 200us, 250us, 280us (przy napięciu: 1.9V, 2.2V, 2.5V)
    *   - przerwanie zegarowe co ~100us + czas obsługin przerwania:
    *     - krótkiego ~50us
    *     - długiego ~100us (detekcja zera)
    *   - szczyt (zero) po ~860us po początku sygnału (~600-660us po detekcji zera)
    */
}

/*
// PWM zero-cross (reinit) interrupt
SIGNAL( INT0_vect ) {
 uint8_t i;

 status[8] = ((PINB & 0x03) << 2) | ((PINC & 0x30) >> 4);

 for (i=0; i<4; i++) {
    if (status[8] & (1 << i)) {
      if (status[i] == 1) {
        // ON
```

```
      PORTC |= ( 1 << i );
      counters[i] = 0;
    } else if (status[i] == 0) {
      // OFF
      PORTC &= ~( 1 << i );
      counters[i] = 0;
    } else {
      // start/init PWM coundown
      counters[i] = status[i];
    }
  }
 }
}
*/

// modbus read/write callback
eMBErrorCode eMBRegHoldingCB( UCHAR * pucRegBuffer, USHORT usAddress, USHORT usNRegs, ⤸
eMBRegisterMode eMode ) {
 // zapytanie dotyczy usNRegs rejestrów począwszy od rejestru usAddress
 uint8_t line = usAddress - SLAVE_BASE_REG - 1;
 uint8_t eline = line + usNRegs;

if (eMode == MB_REG_READ) {
  if (line >= 0 && eline < SLAVE_NUM_REG) {
   for (; line< eline; line++) {
    *pucRegBuffer++ = status[line] >> 8;
    *pucRegBuffer++ = status[line] & 0xff;
   }
   return MB_ENOERR;
  } else if (usAddress == 0xf0ff) {
   *pucRegBuffer++ = VERSION >> 8;
   *pucRegBuffer++ = VERSION & 0xff;
   return MB_ENOERR;
  }
 } else if (eMode == MB_REG_WRITE) {
  if (line < 8) {
   status[line] = *pucRegBuffer++ << 8;
   status[line] |= *pucRegBuffer++;

   if (line < 4 && status[line] > 1)
    status[line] += ON_DELAY;

   // sterowanie liniami on/off
   if (line > 3) {
    if (status[line] == 0) {
     // OFF
     PORTD &= ~( 1 << line );
    } else {
     // ON
     PORTD |= ( 1 << line );
    }
   }
   return MB_ENOERR;
  }
 }
 return MB_ENOREG;
}
```

Sterownik LED RGB / Sterownik linii DC

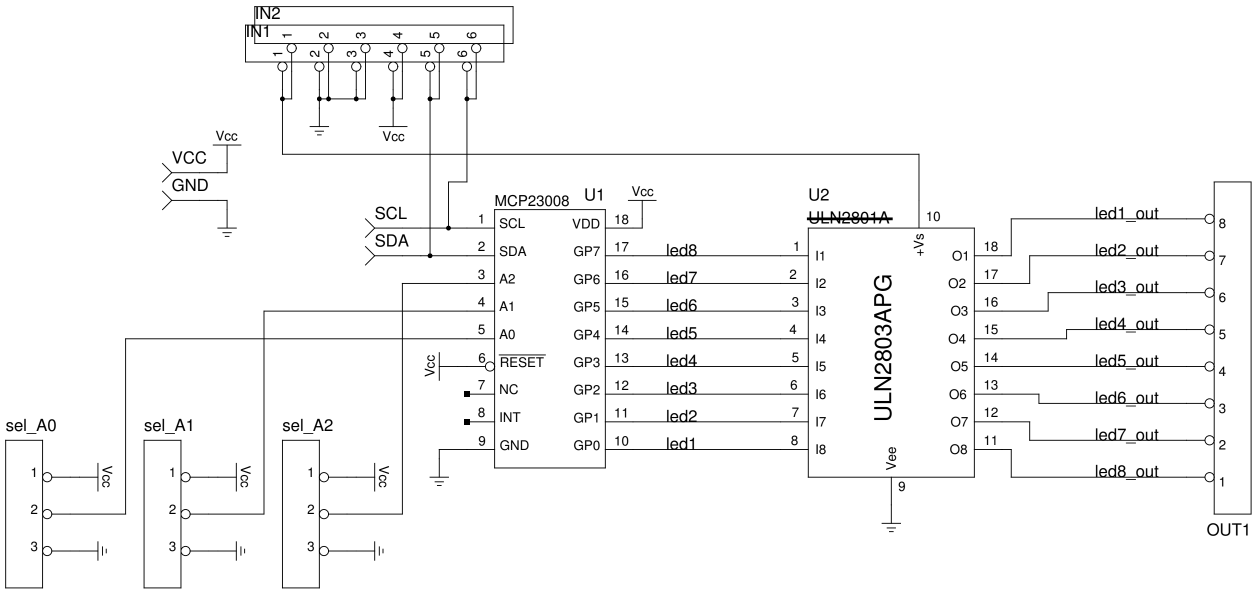
Sterownik oparty jest na magistrali i2c i umożliwia sterowanie wieloma diodami (lub zespołami diód) RGB, w taki sposób że możliwe jest wyświetlanie z ich pomocą płynnej animacji w 12bitowej palecie kolorystycznej (4096 odcieni). Jako master przewidziane zostało zastosowanie kontrolera z interfejsem ethernetowym (do sterowania nim) i i2c (do sterowania modułami wykonawczymi) opartego na procesorze ARM i systemie Linux. Na pojedynczej magistrali i2c jesteśmy w stanie podłączyć bezproblemowo 64 diody LED RGB (np. jakiś wyświetlacz 8x8). Przy założonych 16 poziomach PWM na kolor podstawowy (co daje 12 bitową przestrzeń kolorystyczną), prędkości magistrali i2c 100kHz (ograniczenie PCF8574), koszcie transmisji jednego bajtu wynoszącym 20 cykli zegara i 4*(6+1) urządzeń do wysterowania (cztery linie za multiplekserem po 6 układów + sterowanie multiplekserem) daje nam to częstotliwość odświeżania obrazu: 100000 / (16 * 20 * (6+1)*2) = 22 Hz. Jeżeli chcemy wysterować 64 diody przy zachowaniu rozsądnego odświeżania musimy użyć dwóch magistral i2c lub zamienić PCF8574 na PCA9554 (posiada identyczny pin-out, ale może pracować z szyną 400kHz, koszt transmisji jednego bajtu to 28 cykli). Przy PCA9554 i 4 pod magistralach za multiplekserem odświeżanie wynosić będzie 400000 / (16 * 28 * (6+1)*4) = 31Hz.

Pobierz[2030]

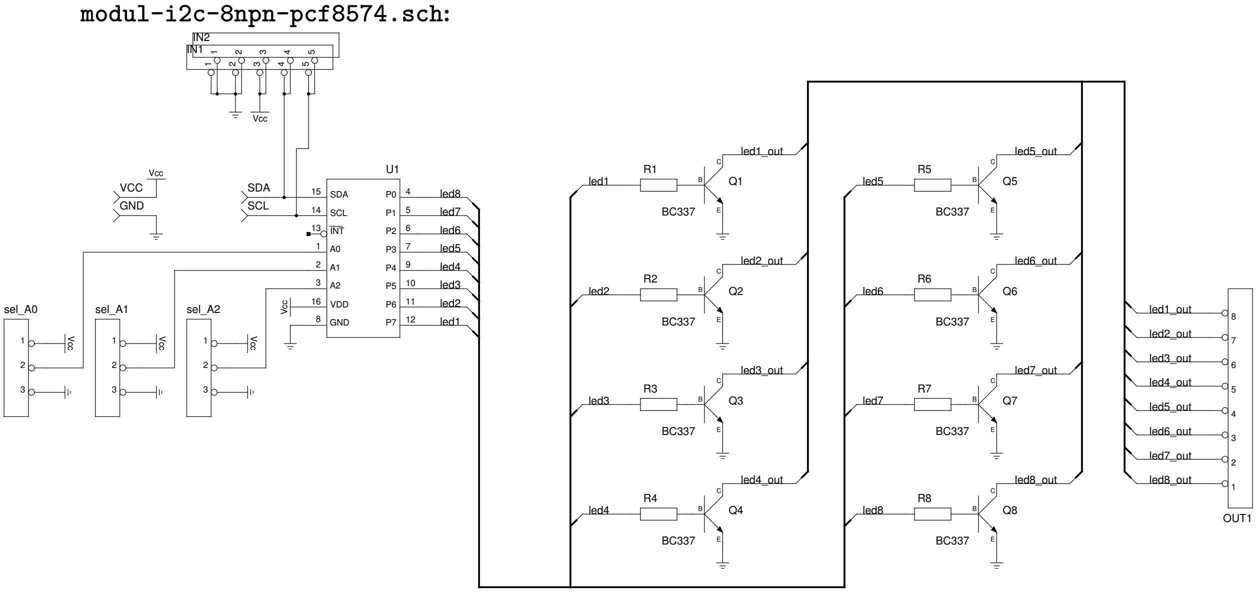
Na projekt sterownika składają się:

- moduł złożony z pojedynczego zestawu 8 tranzystorów NPN (driver prądowego ULN2803APG z 8 układami darlingtona albo realizacja na osobnych tranzystorach NPN) oraz 8mio wyjściowego expandera GPIO szyny I2C (PCF8574/PCA9554 albo MCP23008) wraz z elementami towarzyszącymi:
  - wersja z ULN2803APG i PCF8574/PCA9554 - schemat ideowy[2031](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-pcf8574.pdf`), projekt PCB[2032](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-pcf8574.pcb.pdf`)
  - wersja z ULN2803APG i MCP23008 - schemat ideowy[2033](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-mcp23008.pdf`), projekt PCB[2034](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-mcp23008.pcb.pdf`)
  - wersja z kluczem MOSFET-NPN (opartym o MOSFET i ULN2803APG) i MCP23008 - schemat ideowy[2035](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-mosfet-uln2803a-mcp23008.pdf`), projekt PCB[2036](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-mosfet-uln2803a-mcp23008.pcb.pdf`)
  - wersja z osobnymi tranzystorami i PCF8574/PCA9554 - schemat ideowy[2037](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-8npn-pcf8574.pdf`), projekt PCB[2038](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-8npn-pcf8574.pcb.pdf`)
  - symbol[2039] do wykorzystania przy tworzeniu projektu zbiorczego
- projekt płytki z multiplexerem magistrali i2c: schemat ideowy[2040](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/multiplexer_i2c.pdf`) i projekt PCB[2041](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/multiplexer_i2c.pcb.pdf`)
- program sterujący wyświetlaczem

`modul-i2c-uln2803a-mcp23008.sch:`



`modul-i2c-8npn-pcf8574.sch:`



WYSWIETLACZ.C

```
// wymaga pakietu libi2c-dev

#include <linux/i2c-dev.h>
#include <errno.h>
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>


#define I2C_DEVICE "/dev/i2c-1"
#define MULTIPLEXER_ADDR 0x74

/**
 * OBSŁUGA ZAPISU DO MODUŁÓW I2C Z OBSŁUGĄ MULTIPLEXERA
 */
int file = -1;
int current_i2c_sub_bus = 0;
```

```c
int write_to_i2c(int i2c_sub_bus, int i2c_dev_addr, int i2c_reg_addr, int value) {
 int res;

if (i2c_sub_bus != current_i2c_sub_bus) {
  /// switch multiplexer to i2c_sub_bus

  res = ioctl(file, I2C_SLAVE, MULTIPLEXER_ADDR);
  if (res < 0) {
   fprintf(stderr, "ERROR set device addres to multiplexer (0x%02x): %s\n", MULTIPLEXER_ADDR, ⤸
   strerror(errno));
   return -3;
  }

  res = i2c_smbus_write_byte(file, i2c_sub_bus);
  if (res < 0) {
   fprintf(stderr, "ERROR switch multiplexer (0x%02x) to line %d\n", MULTIPLEXER_ADDR, ⤸
   i2c_sub_bus);
   return -4;
  }

  current_i2c_sub_bus = i2c_sub_bus;
 }

res = ioctl(file, I2C_SLAVE, i2c_dev_addr);
 if (res < 0) {
  fprintf(stderr, "ERROR set device addres to 0x%02x: %s\n", i2c_dev_addr, strerror(errno));
  return -1;
 }

res = i2c_smbus_write_byte_data(file, i2c_reg_addr, value);
 if (res < 0) {
  fprintf(stderr, "ERROR write %d to reg 0x%02x on dev 0x%02x at %d I2C sub-bus\n", value, ⤸
  i2c_reg_addr, i2c_dev_addr, i2c_sub_bus);
  return -2;
 }

return 0;
}

/**
 * OBSŁUGA ZESTAWU MODUŁÓW I2C - inicjalizacja, ustawianie wartości pojedynczego modułu
 */
int init() {
 int i, j, ret = 0;

for (i=4; i<8; i++) {
  for (j=0x20; j<0x23; j++) {
   ret += write_to_i2c(i, j, 0x00, 0x00); // IODIR -> output
   ret += write_to_i2c(i, j, 0x09, 0x00); // GPIO -> disable LED
  }

  if (i != 4)   for (j=0x24; j<0x27; j++) {
   ret += write_to_i2c(i, j, 0x00, 0x00); // IODIR -> output
   ret += write_to_i2c(i, j, 0x09, 0x00); // GPIO -> disable LED
  }
 }

return ret;
}
```

```
#define RED     0x20
#define GREEN   0x21
#define BLUE    0x22
#define OFFSET  0x04
inline int set_leds(int i2c_sub_bus, int i2c_dev_addr, int value) {
 return write_to_i2c(i2c_sub_bus, i2c_dev_addr, 0x09, value);
}

/**
 * OBSŁUGA MATRYCY 5x11 PIKSELI RGB
 */

struct rgb {
 unsigned char r;
 unsigned char g;
 unsigned char b;
};

struct {
 unsigned char line;
 unsigned char offset;

struct rgb    val;
 struct rgb    old;
} moduly[7] = {
 {7, 0, {0x00, 0x00, 0x00}, {0x00, 0x00, 0x00}}, {7, OFFSET, {0x00, 0x00, 0x00}, {0x00, 0x00, ⤸
 0x00}},
 {6, 0, {0x00, 0x00, 0x00}, {0x00, 0x00, 0x00}}, {6, OFFSET, {0x00, 0x00, 0x00}, {0x00, 0x00, ⤸
 0x00}},
 {5, 0, {0x00, 0x00, 0x00}, {0x00, 0x00, 0x00}}, {5, OFFSET, {0x00, 0x00, 0x00}, {0x00, 0x00, ⤸
 0x00}},
 {4, 0, {0x00, 0x00, 0x00}, {0x00, 0x00, 0x00}}
};

#define ROWS 11
#define COLS 5

struct {
 unsigned char modul;
 unsigned char bit;
} matryca[ROWS][COLS] = {
 /*  0 */  { {1, 0x04}, {2, 0x20}, {4, 0x01}, {5, 0x08}, {6, 0x40} },
 /*  1 */  { {1, 0x02}, {2, 0x10}, {3, 0x80}, {5, 0x04}, {6, 0x20} },
 /*  2 */  { {1, 0x01}, {2, 0x08}, {3, 0x40}, {5, 0x02}, {6, 0x10} },
 /*  3 */  { {0, 0x80}, {2, 0x04}, {3, 0x20}, {5, 0x01}, {6, 0x08} },
 /*  4 */  { {0, 0x40}, {2, 0x02}, {3, 0x10}, {4, 0x80}, {6, 0x04} },
 /*  5 */  { {0, 0x20}, {2, 0x01}, {3, 0x08}, {4, 0x40}, {6, 0x02} },
 /*  6 */  { {0, 0x10}, {1, 0x80}, {3, 0x04}, {4, 0x20}, {6, 0x01} },
 /*  7 */  { {0, 0x08}, {1, 0x40}, {3, 0x02}, {4, 0x10}, {5, 0x80} },
 /*  8 */  { {0, 0x04}, {1, 0x20}, {3, 0x01}, {4, 0x08}, {5, 0x40} },
 /*  9 */  { {0, 0x02}, {1, 0x10}, {2, 0x80}, {4, 0x04}, {5, 0x20} },
 /* 10 */ { {0, 0x01}, {1, 0x08}, {2, 0x40}, {4, 0x02}, {5, 0x10} }
};

void display_matryca() {
 int i, j;

for (i=0; i<7; i++) {
  if (moduly[i].val.r != moduly[i].old.r) {
   if (set_leds(moduly[i].line, RED + moduly[i].offset, moduly[i].val.r) == 0 || set_leds(⤸
```

```
   moduly[i].line, RED + moduly[i].offset, moduly[i].val.r) == 0)
    moduly[i].old.r = moduly[i].val.r;
  }
  if (moduly[i].val.g != moduly[i].old.g) {
   if (set_leds(moduly[i].line, GREEN + moduly[i].offset, moduly[i].val.g) == 0 || set_leds(⤸
   moduly[i].line, GREEN + moduly[i].offset, moduly[i].val.g) == 0)
    moduly[i].old.g = moduly[i].val.g;
  }
  if (moduly[i].val.b != moduly[i].old.b) {
   if (set_leds(moduly[i].line, BLUE + moduly[i].offset, moduly[i].val.b) == 0 || set_leds(⤸
   moduly[i].line, BLUE + moduly[i].offset, moduly[i].val.b) == 0)
    moduly[i].old.b = moduly[i].val.b;
  }
 }
}

void prepare_matryca(struct rgb image[ROWS][COLS], int iter) {
 int i, j, m, b;

for (i=0; i<7; i++) {
  moduly[i].val.r = 0;
  moduly[i].val.g = 0;
  moduly[i].val.b = 0;
 }
 for (i=0; i<ROWS; i++) {
  for (j=0; j<COLS; j++) {
   m = matryca[i][j].modul;
   b = matryca[i][j].bit;

   if (image[i][j].r > iter)
    moduly[m].val.r   = moduly[m].val.r   | b;
   if (image[i][j].g > iter)
    moduly[m].val.g = moduly[m].val.g | b;
   if (image[i][j].b > iter)
    moduly[m].val.b  = moduly[m].val.b  | b;
  }
 }
}

/**
 * OBSŁUGA RENDERINGU RAMKI
 */
void render_frame(struct rgb image[ROWS][COLS]) {
 int i;
 int czas;
 struct timeval tv_1, tv_2;

for(i=0; i<12; i++) {
  gettimeofday ( &tv_1, NULL );

  prepare_matryca(image, i);
  display_matryca();

  gettimeofday ( &tv_2, NULL );
   /// 64 ramki po 12 cykli modulacji koloru (PWM) na sekundę => 1000000 / (12*64) = 1300
   czas = 1300 - ((tv_2.tv_sec * 1000000 + tv_2.tv_usec) - (tv_1.tv_sec * 1000000 + tv_1.⤸
   tv_usec));
   if (czas>0) usleep(czas);
 }
}
```

```
/**
 * OBSŁUGA RÓŻNYCH FUNKCJI RENDERINGU
 */
void stroboskop_white() {
 struct rgb image[ROWS][COLS] = {
  { {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1} },
  { {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1} },
  { {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1} },
  { {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1} },
  { {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1}, {0x1, 0x1, 0x1} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} }
 };
 unsigned char strob_cnt, strob_cnt2;
 char strob1, strob2;

while(1) {
  if (strob_cnt%17 == 0) {
   strob2 = 1;
  } else if ((strob_cnt+strob_cnt2)%7 == 0) {
   strob2 = 0;
  }

  if (strob_cnt % 3) {
   strob1 = 1;
   strob_cnt2++;
  } else {
   strob1 = strob2;
  }

  strob_cnt++;

  prepare_matryca(image, strob1);
  display_matryca();

  usleep(70000);
 }
}

void stroboskop() {
 struct rgb image[ROWS][COLS] = {
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} }
 };
```

```
unsigned char strob_cnt, tmp;
 unsigned char strob1, strob2, strob3;
 unsigned char red, blue, yellow, white;

while(1) {
  tmp = strob_cnt%16;
  if (tmp == 6) {
   blue = 0xff;
  } else if (tmp == 8) {
   red = 0x00;
  } else if (tmp == 14) {
   red = 0xff;
  } else if (tmp == 0) {
   blue = 0x00;
  }

  if (strob_cnt%18 == 0) {
   white  = 0x00;
   yellow = 0x00;
  } else if (strob_cnt%27 == 0) {
   white  = 0xff;
  } else if (strob_cnt%9 == 0) {
   yellow = 0xff;
  }

  tmp = strob_cnt%3;
  if (tmp == 1) {
   strob1 = 0xff;
  } else {
   strob1 = 0x00;
  }

  if (tmp == 2) {
   strob2 = 0xff;
  } else {
   strob2 = 0x00;
  }

  if (tmp == 0) {
   strob3 = 0xff;
  } else {
   strob3 = 0x00;
  }

  strob_cnt++;

  image[1][0].r = image[1][1].r = strob1 & red;
  image[1][3].b = image[1][4].b = strob2 & blue;
  image[1][2].r = image[1][2].g = strob3 & (yellow|white);
  image[1][2].b                     = strob3 & white;

  prepare_matryca(image, 0);
  display_matryca();

  usleep(70000);
 }
}

void colors() {
```

```
 struct rgb image[ROWS][COLS] = {
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} }
 };

int a, b, c, i;
 while(1)
  for (a=0; a<13; a++)
   for (b=0; b<13; b++)
    for (c=0; c<13; c++) {
     image[3][0].r = image[3][2].g = image[3][4].b = a;
     image[3][0].g = image[3][2].b = image[3][4].r = b;
     image[3][0].b = image[3][2].r = image[3][4].g = c;
     for (i=0; i<2; i++) render_frame(image);
    }
}

void choinka() {
 struct rgb image_A[ROWS][COLS] = {
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0xe, 0x0, 0xe}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xe, 0xe}, {0x0, 0x0, 0x0} },
  { {0xe, 0xe, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0xe, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0} },
  { {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0xe}, {0x0, 0xf, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} }
 };

struct rgb image_B[ROWS][COLS] = {
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0} },
  { {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} }
 };

struct rgb image_C[ROWS][COLS] = {
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0xe, 0x6, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
```

```
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x6, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0x0, 0x0} },
  { {0xe, 0xe, 0xe}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0} },
  { {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0}, {0x0, 0xf, 0x0} },
  { {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0}, {0x0, 0x0, 0x0} }
 };

int i;

while(1) {
  for (i=0; i<32; i++)
   render_frame(image_A);
  for (i=0; i<32; i++)
   render_frame(image_B);
  for (i=0; i<32; i++)
   render_frame(image_C);
 }
}

main(int argc, char *argv[]) {
 int ret, i;

file = open(I2C_DEVICE, O_RDWR);
 if (file < 0) {
  fprintf(stderr, "ERROR open I2C device file (%s): %s\n", I2C_DEVICE, strerror(errno));
  return -1;
 }

if ( init() < 0 ) {
  fprintf(stderr, "ERROR init I2C devices\n");
  return -1;
 }

if(argc == 2) {
  if (*(argv[1]) == '1') colors();
  if (*(argv[1]) == '2') stroboskop();
  if (*(argv[1]) == '3') stroboskop_white();
  if (*(argv[1]) == '9') choinka();
 }
}
```

Przełącznik sygnałów audio

Chcąc używać jednego zestawu głośnikowego z kilkoma źródłami sygnału zachodzi potrzeba zastosowania przełącznika takich sygnałów. Niestety ciężko (w odróżnieniu np. od przełączników hdmi) kupić odpowiedniego gotowca. Zamieszam tutaj prosty projekt dwu kanałowego switcha sygnałów audio (jeżeli potrzebny jest switch 6-ścio kanałowy to wystarczy pod wspólne sterowanie podłączyć 3 takie moduły): schemat ideowy[2042](PDF: http://www.opcode.eu.org//usage_and_config/smart_home/audio_switch/switch_audio.pdf). W zależności od potrzeb możliwe jest stosowanie na wejściach, wyjściach mogą być stosowane gniazda minijack, chinch lub kable przyłączeniowe.

Projekt w fazie koncepcyjnej - wymaga przeprowadzenia testów poprawności działania

Niestabilizowany zasilacz buforowy

Przedstawiam projekt niestabilizowanego zasilacza buforowego. Układ pomyślany został to pracy jako zasilacz centralny zasilacz systemów "elektronicznego domu" (systemy sterowania, system alarmowy, systemy komunikacyjne - router wifi z switchem, telefoniczne - telefony SIP, komputerowe - laptop, diodowe oświetlenie awaryjne).

W urządzeniach elektronicznych możemy wyróżnić następujące popularne standardy zasilania DC:

- napięcia precyzyjne (mogą trafiać bezpośrednio do układów scalonych, itd):
  - 3.3V
  - 5V
  - 9V (nie wiem czy używane bezpośrednio czy tylko do wytworzenia innych napięć)
  - 12V (dyski, wentylatory, ale często tylko do wytworzenia innych napięć)
- napięcia "brudne" (z sporą tolerancją - nie używane bezpośrednio, a jedynie do wytworzenia innych napięć):
  - 18V (19V)
  - 24V (16V-30V), w logice 24V napięcie >= 15V przyjmuje się za jedynkę
  - 48V (36V-72V)

Akumulatory w zależności od stopnia naładowania oferują nam napięcia z przedziału: 11V - 13.8V (12V), 16.5V - 20.7V (18V), 22V - 27.6V (24V).

W projekcie zdecydowałem się na realizację zasilacza na 18V (czyli zasadniczo 16-21V). Napięcie 18V zostało podyktowane łatwością uzyskania wszystkich standardowych napięć poprzez przetwornice DC/DC - w przypadku 12 lub 24V ciężko byłoby uzyskać te napięcia (wymagane przetwornice up-down). Można rozważyć modyfikację tego projektu aby uzyskiwać z niego 12V (dokładniej to coś z przedziału 10.5-14.5). Wtedy (przynajmniej dla napięcia 12V) należałoby korzystać z przetwornic izolowanych.

Projekt zakłada:

- centralny zasilacz (pseudo-)buforowy z akumulatorami 18V
- ładowanie akumulatorów niezależne od zasilania odbiorników (jeden mostek + zestaw kondensatorów zasila szynę DC, drugi mostek zasila układ ładowania akumulatorów, oba układy mają wspólną masę)
- przy pracy sieciowej napięcie 22V, po przejściu na akumulatory 18V spadek do 20.7V, co jest to jednocześnie sygnalizacja (dla urządzeń odbiorczych) trybu pracy akumulatorowej
- napięcia precyzyjne uzyskiwane przez lokalne zasilacze dc-dc step-down

Projekt został zarzucony na etapie koncepcyjnym na rzecz zastosowania gotowego impulsowego zasilacza buforowego 12V wyposażonego w wymagane zabezpieczenia zwarciowe, przeciążeniowe i przeciw głębokiemu rozładowaniu. Nie zostały wykonane testy prezentowanych rozwiązań. Zaletą prezentowanego tutaj rozwiązania powinny być mniejsze straty (brak stabilizacji), ale czy tak byłoby rzeczywiście ciężko ocenić (straty na trafo, brak możliwości bezpośredniego wykorzystania nietypowego niestabilizowanego napięcia 18 V vs zasilacz impulsowy i 12V napięcie które możemy podłączyć bezpośrednio - bez kolejnej przetwornicy DC/DC do części urządzeń).

Zamieszczam schemat ideowy zasilacza[2043](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/niestabilizowany_zasilacz_buforowy/./niestabilizowany_zasilacz_buforowy.pdf`), a także kilka propozycji[2044](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/niestabilizowany_zasilacz_buforowy/elementy_podzasilaczy.pdf`) "podzasilaczy" i płytki dystrybucji napięć wyjściowych[2045](PDF: `http://www.opcode.eu.org//usage_and_config/smart_home/niestabilizowany_zasilacz_buforowy/./podlaczenie_zasilania.pdf`).

Dobór transformatora i kondensatorów filtrujących

Pierwszą kwestią jest charakterystyka transformatora czyli jak wygląda funkcja napięcia od pobieranego prądu. W przypadku transformatorów sztywnych (np. transformatory troidalne) jest to funkcja niemal stała (przy zerowym obciążeniu napięcie wynosi około 110% napięcia nominalnego), w przypadku transformatorów miękkich napięcie bez obciązenia może być nawet ponad 160% napięcia nominalnego. Praktycznie zawsze trzbea wybrać transformator o większej mocy niż wynikałoby to z nominalnego obciążenia (chociażby ze względu na warunki chłodzenia czy też prądy rozruchowe).

Napięcie nominalne transformatora to tzw. napięcie skuteczne przy obciążeniu 100%. Amplituda jest pierwiastek z 2 razy większa od wartości skutecznej. Po wyprostowaniu na podłączonym kondensatorze będziemy mieli napięcie odpowiadające amplitudzie pomniejszonej o spadki napięcia na dwuch diodach (w przypadku prostownika dwupołówkowego - mostka prostowniczego) - zależą one od płynącego prądu oraz diody i są zazwyczaj od 0.6V do 1.5V.

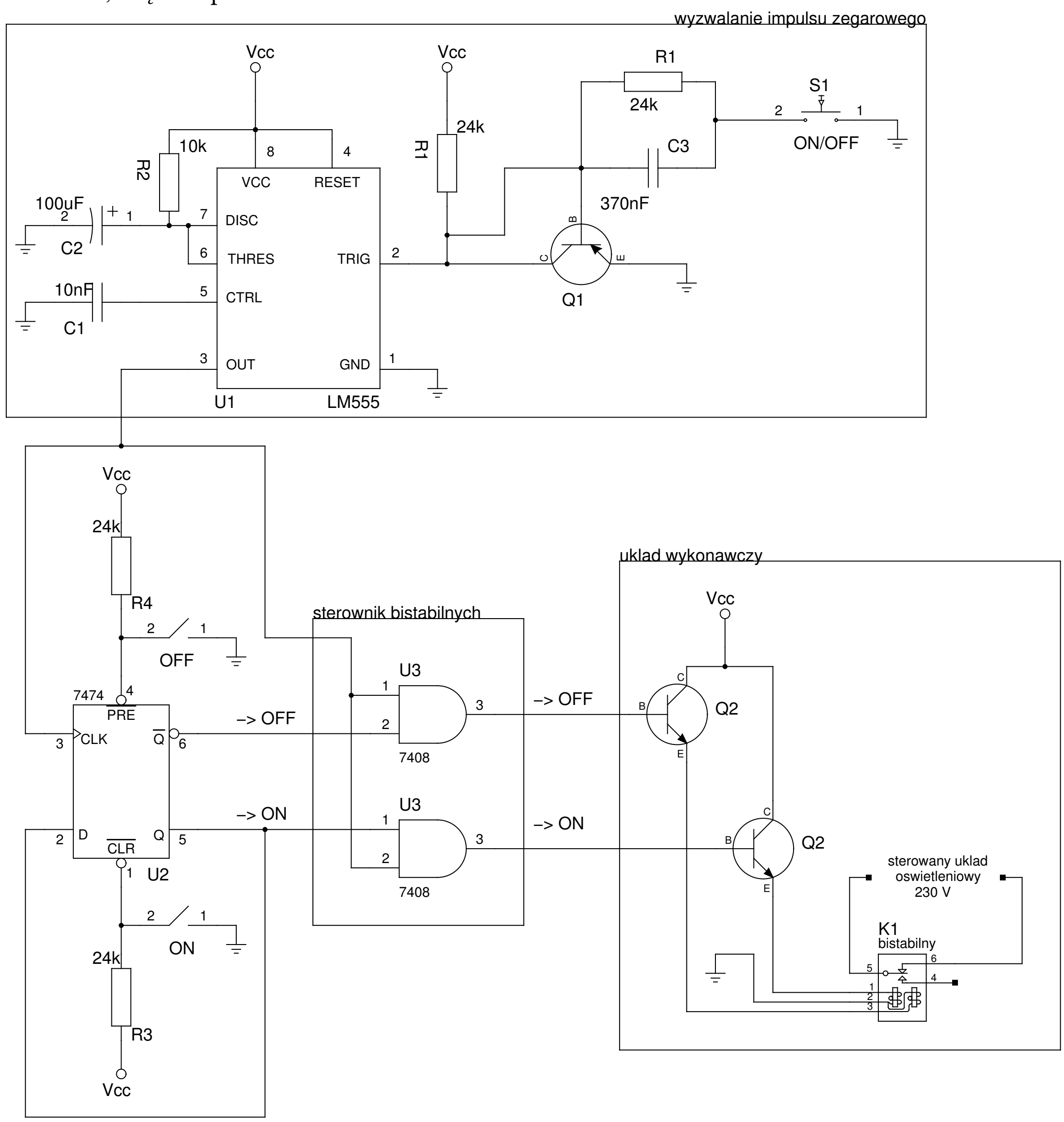
Pojemność kondensatora dobiera się w zależności od pobieranego prądu iżądanej wartości tętnień i wynosi ona dla prostownika dwupołówkowego prąd * 10ms / wartość tętnień.

Ewolucja koncepcji projektów inteligentnego domu

Artykuł ma na celu prezentację rozwoju prezentowanych w serwisie projektów związanych z tzw. inteligentnym domem, czyli różnego rodzaju układów sterowania i automatyki domowej, a także przybliżyć interesujące z jakiś względów rozwiązania, które nie znalazły miejsca w aktualnych projektach.

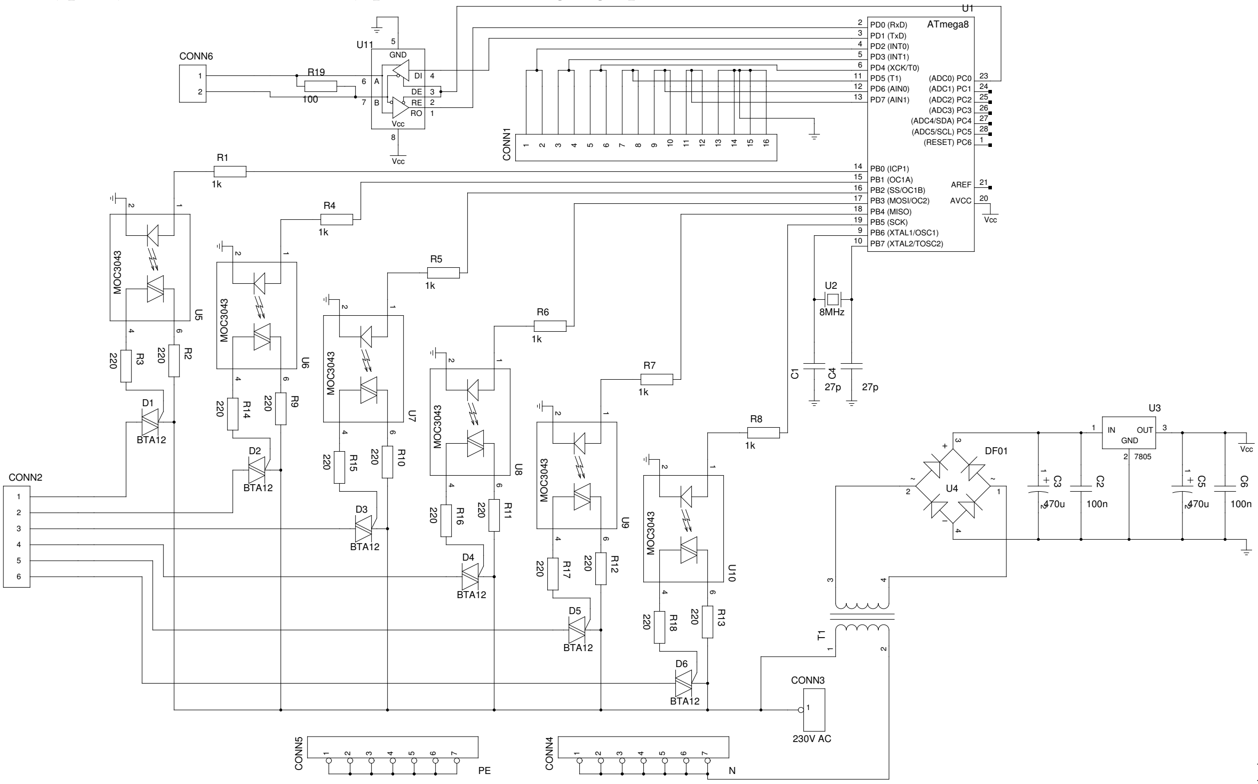
Rozwiązania oparte na układach cyfrowych małej skali integracji

Pierwszym z projektów automatyki domowej był impulsowy sterownik oświetlenia w oparciu o układy cyfrowe serii 74 (TTL) oraz przekaźniki bistabilne. Umożliwiał on także zastosowanie zwykłych przekaźników (po wyeliminowaniu części układu odpowiedzialnego za impulsowość sterowania. Głównym problemem takiej realizacji było to iż układ wymagałaby użycia wielu elementów (dla sterownika 6-cio kanałowego: 6x NE555, 3x 4747, 3x 4708 oraz wiele elementów biernych) a co za tym idzie rozbudowanego i zajmującego sporo miejsca PCB. Zamieszczam tutaj powstały wtedy projekt takiego sterownika, jednak należy pamiętać że projekt nigdy nie doczekał się realizacji czy nawet testów, więc ma prawo nie działać.



Mikrokontrolery AVR

Celem zmniejszenia rozmiarów oraz uproszczenia budowy zdecydowałem się na zastosowanie mikrokontrolera ATmega8. Układ taki miał też większe możliwości (w szczególności prostsze zarządzanie zdalne z wykorzystaniem RS-485 lub RS-422A). Równocześnie w miejsce przekaźników zdecydowałam się także na zastosowanie zestawu optotriak + triak. Pierwotna wersja sterownika posiadała na swoim PCB autonomiczne zasilacze sieciowe (niezależne dla każdego sterownika). Podyktowane zostało to rozwiązaniem zasilania instalacji oświetleniowej (układ zabezpieczeń nad i różnicowoprądowych) nie bardzo pozwalającym na centralne zasilenie (wyłączenie jego zabezpieczenia rozwalałoby cały system, który obecnie podzielony jest na strefy z niezależnymi zabezpieczeniami). Został zrealizowany jeden prototypowy sterownik (działający p;o dziś dzień) wg tego projektu.



Porównanie z koncepcją realizacji na elementach dyskretnych pokazuje przewagę rozwiązań opartych na mikrokontrolerze (gdzie elektronika na układach dyskretnych pełni tylko pomocniczą funkcję). Jest ona spowodowana ogromnymi możliwościami mikrokontrolerów oraz drastycznym uproszczeniem budowy układu, zmniejszeniem rozmiarów, a często także kosztów w stosunku co do realizacji analogicznych układów cyfrowych w oparciu o bramki, przerzutniki itp..

W drugiej wersjach sterownika scentralizowane zostało ich zasilanie, pojawiła się funkcja regulacji jasności wykonana w technologii Pulse Width Modulation[2046] dla jednej linii, a także (obok typowej "płaskiej" wersji) została zaprojektowana wersja dwupoziomowa[2047] do umieszczenia w typowej obudowie na szynę DIN.

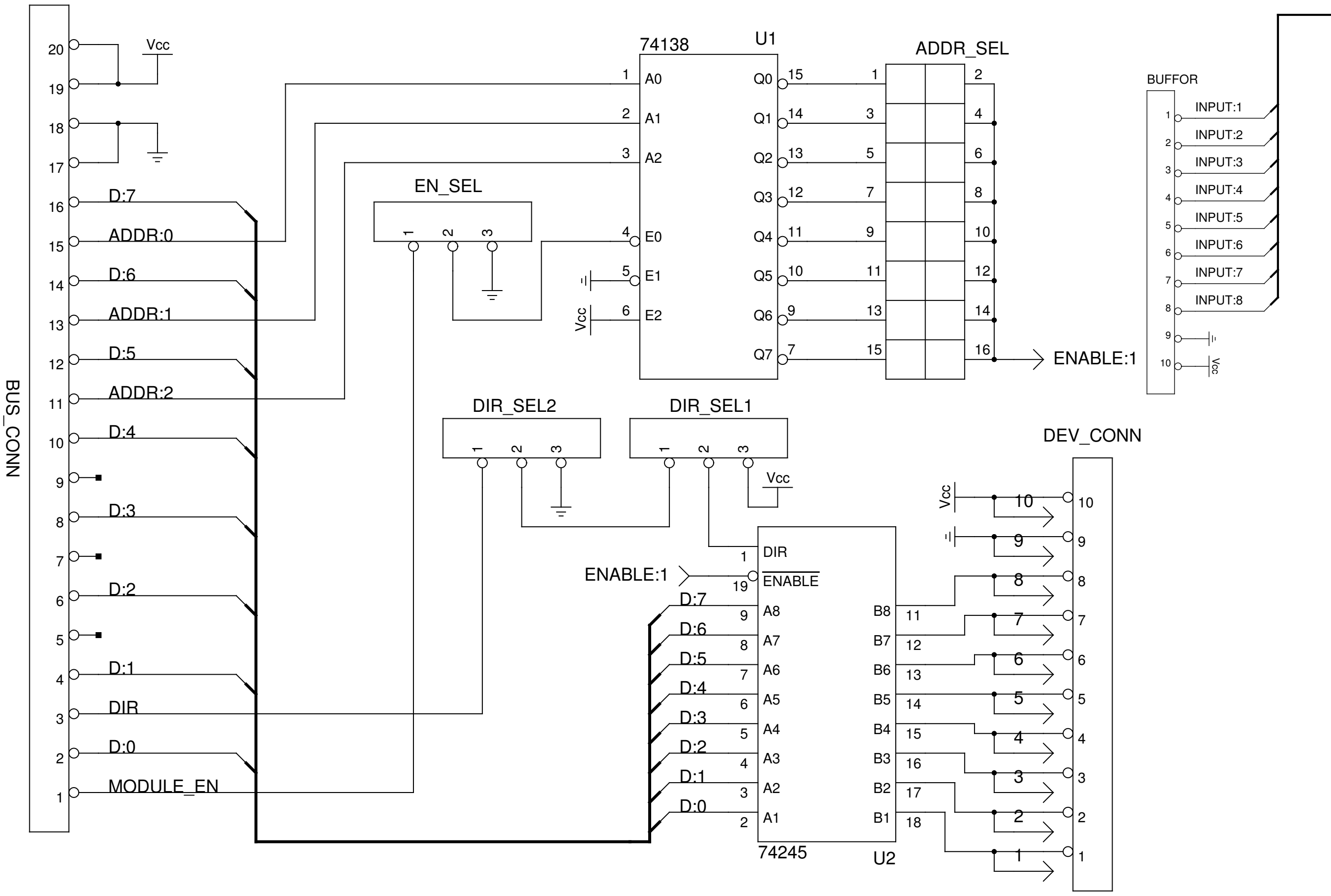
W trzeciej wersji sterowanie PWM zostało rozszerzone na wszystkie linie, a dzięki przeniesieniu obsługi linii sterujących z przełączników do innego modułu (pozostało tylko sterowanie po RS-485) zwiększono liczbę linii do 15tu. W (nie zrealizowanej) erracie tej wersji wprowadzone zostały także opcjonalne układy gasikowe (umożliwiające poprawną pracę z obciążeniem indukcyjnym - świetlówki) i dławiki. Z wersją trzecią zmieniona została także sama metoda projektowania - wykorzystana została powtarzalność układu do hierarchizacji projektu - patrz hierarchiczne schematy w gEDA.

Sterownik w trzeciej wersji zaprojektowany został do wykorzystania w 19" systemie rozdzielnic elektrycznych opisywanym w elektroniczny dom. Sterownik przystosowany był do montażu (max po dwa moduły sterownika) w obudowach rack 19" 3U/250mm mających od frontu szyna DIN do montażu zabezpieczeń (opcjonalnie można też umieścić szynę DIN do montażu dodatkowej aparatury prostopadle do frontowej wewnątrz obudowy panelu). Sterownik montowany powinien być na przestrzeni środkowego 1U. Sterownik może być montowany w pod-obudowie posiadającej otwarte wyprowadzenie kabli podłączeniowych od tyłu (do złączek szynowych montowanych na tylnej ścianie szafy) i zdejmowalną górną pokrywę. Na przedniej maskownicy aparatury modułowej można zamontować dodatkowo diody kontrolne (zasilanie sterownika + napięcie kontrolne sterownika + stan każdej linii) oraz gniazdo RJ-11 dla RS485 i gniazdo zasilania DC. Moduł sterownika odpowiedzialny za obsługę sygnałów wejściowych jest montowany w osobnej obudowie 1U, może być także zintegrowany z centralką alarmową itp urządzeniami. Zobacz także zdjęcia prototypu[2048]

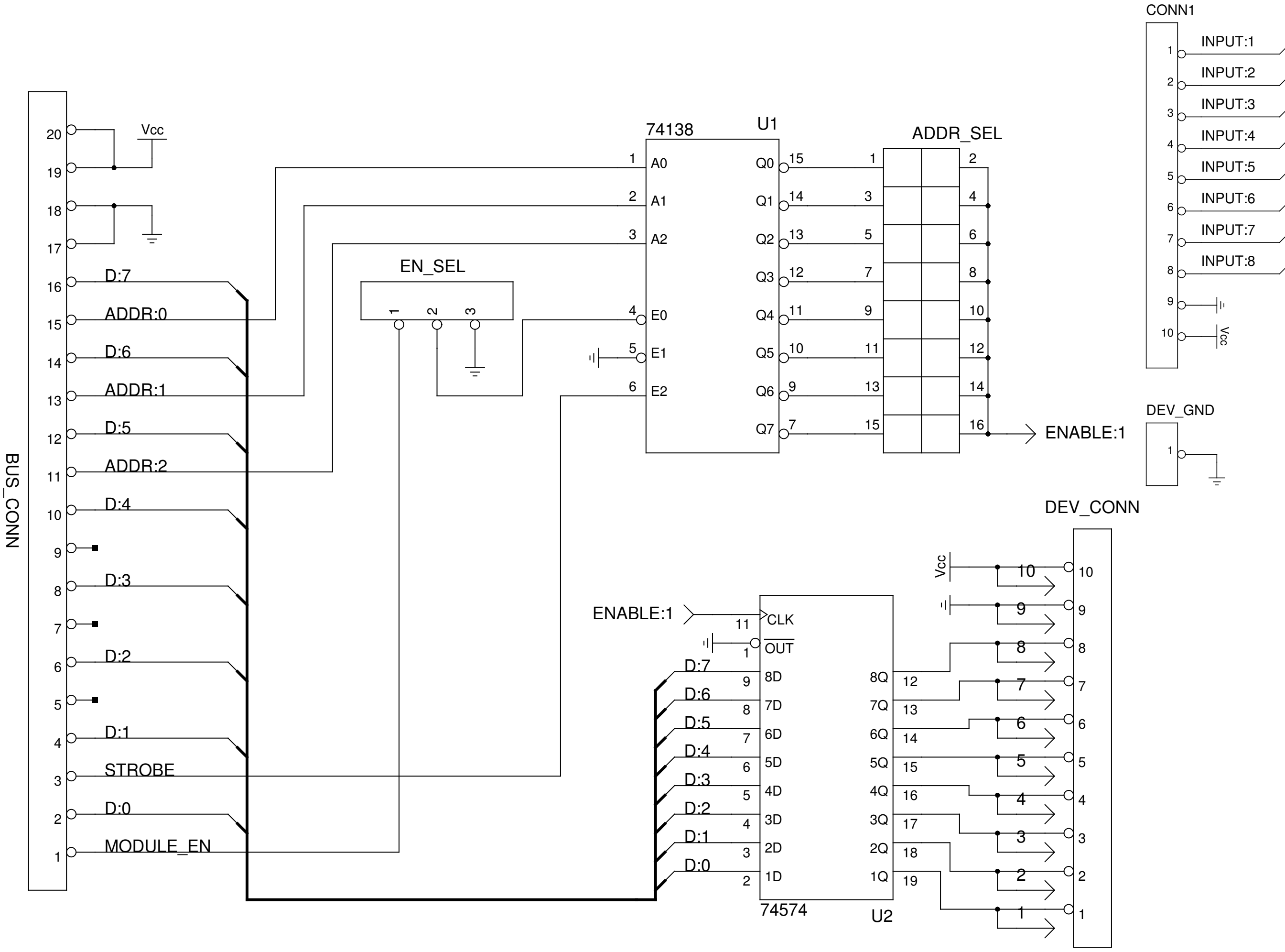
Centralka

Jak wspomniałem funkcje obsługi wejść (przełaczników) w wrsji trrzeciej zostały przekazane do innego modułu - była nim centralka alarmowa, oparta na mikrokontrolerze Atmega16. Projekt oparty był na modułach podłączanych do mikrokontrolera poprzez magistralę równoległą utworzoną z portu B mikrokontrolera (8 bitów danych) i części portu A (6 bitów adresowych). Detekcja adresu w modułach realizowana była w oparciu 3 najmłodsze bity (i sygnał zezwolenia) adresu na układzie 74hc183. 3 najstarsze bity adresu mogłyby·ć wykorzystane przez switch magistrali oparty także na układzie 74hc183 do generowania sygnału zezwolenia.:

Moduł wejścia-wyjścia (oparty na buforze dwukierunkowym) i bazujący na nim moduł wejść optoizolowanych (4 zwykłe, 4 z filtrem rc):



duł wyjścia (oparty na rejestrze 8 bitowym) i bazujący na nim moduł kluczy NPN oraz NPN (sterowanie układem od strony masy) + P-MOSFET (sterowanie układem od strony bieguna dodatniego):



Centralka - wnioski po-wdrożeniowe

- nie najlepszym pomysłem jest bezpośrednie przyłączanie przewodów wyjściowych (w wdrażanym rozwiązaniu były one zakończone wtykami RJ45 podłączanymi do panelu krosowego) do modułów ze względu na konieczność podpięcia przewodów od jednego urządzenia do kilku modułów wejściowych (np. od czujki p.poz z termometrem 1-wire były to moduły: wyjść mosfetowych, filtrowanych wejść optoizolowanych, wejściowo-wyjściowy)
  - należy rozważyć zastosowanie modułów podłączania kabli - z złączami śrubowymi do podłączania przewodów i/lub bezpośrednio dolutowanymi przewodami oraz złączami taśmowymi IDC do podłączania kilku (np. czterech lub ośmiu modułów centralki), dodatkowo w ramach tego złącza taśmowego należy przewidzieć dystrybucję dodatkowych zasilań 5V, 9V, 12V na potrzeby podłączanego modułu centralki
- pewną uciążliwością charakteryzuje się brak możliwości podpięcia poszczególnych elementów na modułach (np. wejść optoizolowanych) do wspólnych mas/zasilań bez lutowania dodatkowych przewodów na tych modułach
  - należy rozważyć bezpośrednie wprowadzenie tych linii na złącze sygnałów wejściowych i zapewnienie jakiegoś konfigurowalnego mechanizmu podłączania do szyn mas/zasilania na module przyłączeniowym
- problematyczne było podłączanie zewnętrznych magistral RS485
  - należy przewidzieć stosowne miejsca w module wejściowym i piny dla magistrali RS485 w złączu taśmowym wejść lub w złączu taśmowym magistrali
- problem stanowił też brak standaryzacji wielkości modułów i mechanizmu ich mocowania do wspólnej płyty
- można rozważyć realizację magistrali w oparciu o złącza krawędziowe na płytkach i płytę bazową zamiast obecnie stosowanego 20 pinowego złącza taśmowego
- można rozważyć wykorzystanie złącz RJ45 umieszczanych bezpośrednio na module do łączenia modułów umieszczanych wewnątrz szafki z centralką (np. sterowników oświetlenia) - w tym wypadku możliwe byłoby pominięcie w ich podłączaniu panelu krosowego przy zachowaniu jednolitego standardu podłączeń
- przyjęta koncepcja magistrali z rejestrami i buforami okazała się trudna do pogodzenia z stosowaną koncepcją

linii sterowania jasnością gdzie na tym samym przewodzie mamy sygnał wyjściowy o możliwości regulacji i wejściowy zadający regulację - rozwiązanie to wymaga możliwości zmiany kierunku z zapamiętaniem wartości wyjściowej i rezystorami podciągającymi w wariancie wejściowym
  - możliwe jest przeznaczenie na to bezpośrednich wejść mikrokontrolera, jednak ogranicza to znacznie ilość takich linii
  - innym rozwiązaniem jest skonstruowanie specjalnego modułu magistralowego realizującego tą funkcjonalność
  - możliwe jest też rozwiązanie tego problemu przez zmianę koncepcji linii sterowania jasnością i rozdzielenie funkcji wejściowej od wyjściowej
  - jeszcze innym (i chyba najprostszym) jest wydelegowanie tej funkcjonalności do modułu mikrokontrolerowego komunikującego się po RS485, wariant taki umożliwiłby także sygnalizację nastrojenia jasności (a nie tylko możliwego kierunku regulacji) z wykorzystaniem PWM
- interesująca koncepcją jest stosowanie rozproszonych systemów wejść wyjść opartych na projekcie manipulatora alarmowego, gdzie np. zamiast LCD podłączone byłyby dodatkowe linie na bezpośrednie przyciski, a sterowanie dowolną linią mogłoby odbywać się z klawiatury
- warto rozważyć przy realizacji w centralce tylko jednego portu UART rozdzielenie linii nadawczej i odbiorczej - zapewnia to niezależność transmisji od centralki w stosunku co do transmisji do niej - komunikacja z układami wykonawczymi centralki tylko za jej pośrednictwem

Aktualna wersje projektów (w szczególności dostosowane do wymogów projektowych opisanych w mikrokontrolery) znajdują się w dziale z projektami związanymi z inteligentnym domem: sterownik oświetlenia, centralka alarmowa. W pierwotnych założeniach przewijało się stosowanie magistral równoległych w niektórych modułach (np. sterownikach tranzystorowych) do zwiększania ilości wejść/wyjść. Pomysł ten nie został jednak zrealizowany ze względu na powyżej opisane trudności, zamiast tego zostały zastosowane moduły i2c podłączone do głównego sterowniki (w szczególności moduł oparty na mikrokontrolerze pełniącym funkcję slave'a i2c). W ogólności takie moduły można stosować jako (zaawansowane) rozszerzenia wejść wyjść mikrokontrolera (zaletą standaryzacja, niższy koszt na pin niż dedykowane układy i2c, oszczędność 3 pinów w stosunku co do wersji modbusowej).

Komunikacja poprzez RS485

Wraz z użyciem mikrokontrolerów pojawiła się idea zdalnego sterowania poprzez RS485[2049] z wykorzystaniem wbudowanych układów UART[2050]. Wiązało się to z koniecznością przyjęcia/stworzenia jakiegoś protokołu do przekazywania komend. W pierwszych rozwiązaniach stosowany był prosty binarny format komunikatów - komunikaty były 4 bajtowe, a dwa pierwsze bity oznaczały typ bajtu w komunikacie (identyfikator urządzenia, identyfikator linii, komenda, odpowiedz). Implementacja tego protokołu zawarta jest w my_rs485_binary.c, a przykład użycia w sterownik_oswietlenia_v2.0.c. Rozwiązanie to ograniczało przestrzeń dostępnych adresów pod-adresów i komunikatów do 6 bitów, a także uniemożliwiało w prosty sposób przekazywania argumentu do komendy (co stało się potrzebne przy sterowaniu jasnością większej ilości linii). Niedogodnością była też niemożność prostego używania terminala (/dev/ttySx) do sterowania urządzeniami.

Kolejnym rozwiązaniem była komunikacja tekstowa ASCII w postaci: `@adres#subadres#komenda#parametr;` lub `@adres#subadres#komenda;`, gdzie adres, subadres, komenda, parametr są liczbami z zakresu 0-255 zapisanymi dziesiętne. Implementacja tego protokołu zawarta jest w my_rs485_ascii.c, a przykład użycia w sterownik_oswietlenia_v3.1.c.

W obu wypadkach całość dekodowania i (typowo) wykonywania komendy odbywała się w ramach przerwań związanych z otrzymaniem znaku z portu szeregowego. Niestety ze względu na konieczność stosowania znacznych opóźnień między-znakowych w komunikacji RS485, wynikłą z potrzeby dostosowania się do najwolniejszych modułów, które muszą mieć czas na analizę otrzymanego komunikatu system ulega stosunkowo łatwo zamulaniu. W związku z tym rozważana była zmiana koncepcji odbioru komunikatów z stosowanej analizy w trakcie odbioru na szybki odbiór analizę po jego zakończeniu i potwierdzenie wykonania po odebraniu komendy, ale wymagałoby to stosowania dodatkowego timeoutu na oczekiwaniu na potwierdzenie wykonania.

Kolejnym problemem była chęć umożliwienia nadawania komend (a nie tylko odpowiadania na nie) wielu urządzeniom. Wymusiła ona konieczność stosowania jakiejś formy arbitrażu. Zastosowane w pierwotnej wersji centralki alarmowej indywidualne linie zezwolenia na nadawanie nie zdały niestety egzaminu. Rozważane były m.in.:

- zastosowanie protokołu żetonowego z żetonem przekazywanym przy pomocy komendy
- system oparty na centralnym masterze odpytujący każde inne urządzenie o to czy ma coś do nadania
- system z stałym oknem czasowym na nadawanie (a nie rozpoczęcie nadawania) zadawanym indywidualna linią zezwolenia
- jakieś usprawnienia obecnego systemu niedopuszczające do dyskryminacji i zagładzania urządzeń

Komunikacja binarna i switche - koncepcje niezrealizowane

W efekcie opisanych powyżej problemów z opóźnieniami w transmisji zrodził się pomysł **wykorzystania echa (odsyłania odebranego bajtu) do** weryfikacji odbioru komunikatu, jej wykonania oraz odbioru odpowiedzi, a przede wszystkim do **sterowania prędkością nadawania** (czekamy na potwierdzenie odbioru i przetworzenia poprzedniego bajtu). Równocześnie nastąpił powrót do komunikatów binarnych (mniejszy narzut na transmisję oraz prostsza analiza komendy). Komunikat miał składać się z 6 bajtów:

- pierwszy bajt - adres urządzenia (0-32) 0b000x xxxx
- drugi bajt - komenda (0-32) 0b001x xxxx
- 3+4. dana 1 (subadres)
  - młodsza polowa bajtu: 0b0100
  - starsza polowa bajtu: 0b0101
- 5+6. dana 2 (wartość)
  - młodsza polowa bajtu: 0b0110
  - starsza polowa bajtu: 0b0111

Kodowanie typu bajtu w każdym z bajtów poprawia czytelność protokołu, ułatwia interpretację komunikatu (a także umożliwia detekcję zagubionego bajtu), a także restart transmisji (znacznik pierwszego bajtu).

Algorytm odbioru komunikatu zapisany w pseudo-c:

```
uint8_t rd, cmd, d1, d2;
uint8_t mask = MY_ADDRESS;
#ifdef USE_INTERPUT
procedura_obslugi przerwania() {
        rd = UDR;
        // zakladamy ze nadawca czeka na potwierdzenie wiec nie bedzie nadawl gdy
↪  przetwarzamy
        odblokuj_przerwania();
#else
while (1) {
        rd = wait_for_data();
#endif
        if ((rd & mask) == rd) {
                switch (mask) {
                        /// adres
                        case MY_ADDRESS:
                                mask = 0x20 | 0x1f;
                                cmd = 0x00;
                                break;
                        /// komenda
                        case 0x20 | 0x1f:
                                mask = 0x40 | 0x0f;
                                cmd = rd &0x1f;
                                break;
                        /// dana 1
                        case 0x40 | 0x0f:
                                mask = 0x40 | 0x10 | 0x0f;
                                d1 = rd &0x0f;
                                break;
                        case 0x40 | 0x10 | 0x0f:
                                mask = 0x60 | 0x0f;
                                d1 = d1 | ( ( rd << 4 ) & 0x0f);
                                break;
                                cmd = execute_cmd1(cmd, d1, &d2); // funkcja wykonuje
↪  komendę która ma odpowiedź i zapisuje ją w d2, jeżeli się udało (była taka komenda
↪  zwraz 0xff, w przeciwnym razie cmd)
                        /// dana 2
                        case 0x60 | 0x0f:
                                mask = 0x60 | 0x10 | 0x0f;
                                if (cmd == 0xff) {
                                        send( 0x60 | (d2 &0xf) );
```

```
                                } else {
                                        d2 = rd &0x0f;
                                }
                                break;
                        case 0x60 | 0x10 | 0x0f:
                                mask = MY_ADDRESS;
                                if (cmd == 0xff) {
                                        send( 0x60 | ((d2<<4) &0xf) );
                                } else {
                                        d2 = d2 | ( ( rd << 4 ) & 0x0f);
                                        execute_cmd2(cmd, d1, d2); // funkcja wykonuje
↪ komendę która nie ma odpowiedzi
                                }
                                break;
                }
                if (cmd != 0xff)
                        send(rd);
        } else {
                mask = MY_ADDRESS;
        }
}
```

Algorytm nadawania komunikatu zapisany w pseudo-c:

```
send_msg(uint8_t addr, uint8_t cmd, uint8_t d1, uint8_t d2) {
        uint8_t buf[6], i, ret, reply = 0;
        buf[0]=addr;
        buf[1]=0x20 | cmd;
        buf[2]=0x40 | (d1 & 0x0f);
        buf[3]=0x40 | 0x10 | ((d1<<4) & 0x0f);
        buf[4]=0x60 | (d2 & 0x0f);
        buf[5]=0x60 | 0x10 | ((d2<<4) & 0x0f);
        for (i=0; i<6; i++);
                send(buf[i]);
                ret = wait_for_data_with_timeout();
                if (i == 4 && d2 == 0xff) {
                        if (ret & 0xf0 ! = 0x60) {
                                // błąd transmisji
                                return i - 20;
                        }
                        reply = ret & 0xf;
                } else if (i == 5 && d2 == 0xff) {
                        if (ret & 0xf0 ! = 0x70) {
                                // błąd transmisji
                                return i - 20;
                        }
                        reply = reply | ( (ret<<4) & 0xf );
                } else if (ret != d[i]) {
                        // błąd transmisji
                        return i - 10;
                }
        }
        return reply;
}
```

Niestety nie rozwiązywało to problemu rywalizacji o dostęp do medium transmisyjnego, rozważano w tej kwestii:

- użycie `0b111 | ADRES` jako tokenu uprawniającego do nadawania
- użycie linii open-collector informacja że właśnie nadaje
- rezygnacja z magistrali na rzecz łącz punkt(switch)-punkt(slave):

- gwiazda z hubem/switchem (mniej lub bardziej inteligentnym) wybierającym (w oparciu o zgłoszenia na dedykowanej linii) tego kto aktualnie nadaje do wszystkich pozostałych (ew. może także rozpoznawać adresy i nadawać tylko na właściwe porty), przykładowe realizacje takiego huba: w oparciu o sterowanie z mikrokontrolera[2051] i bez mikrokontrolera sterującego[2052]
- wariant rs_ttl full-duplex: nadawanie i odbiór poprzez bufor (74hc125) do wskazanego z użyciem dekodera (74hc138, zerowe wyjście nie używane) slave'a
- wariant rs_485 full-duplex: nadawanie i odbiór poprzez trancriver (sn75176) do wskazanego z użyciem dekodera (74hc138, zerowe wyjście nie używane) slave'a
- wariant rs_485 half-duplex: nadawanie do wszystkich (osobne sn75176) słuchanie tylko z wybranego z użyciem dekodera (74hc138, zerowe wyjście nie używane) slave'a; nadajniki w sn75176 (DE) sterowane z 74hc238 (zerowe wyjście nie używane), odbiorniki (RE) sterowane z 74hc138 (zerowe wyjście nie używane), wejście E i E dekoderów podłączone do linii sterującej zapis/odczyt
- linia sygnalizująca chęć nadawania wprowadzona na 8in encoder (74hc148, zerowe wejście podłączone do masy) i do mastera decydującego komu dać prawo nadawania uC
- sygnał zgody na nadawanie (włączenia danego odbiornika) przekazywany do urządzenia slave w formie linii open-collector (hc7407)
  - stan niski nie ustawiony przez slave informuje o tym że można nadawać
  - stan niski nie ustawiony przez mastera oznacza prośbę o transmisję odpowiedzi (aby móc wykryć zgodę na transmisję odpowiedzi musi on być przerywany celem odczytu stanu linii, zatem nie ma potrzeby osobnej zgody na transmisje odpowiedzi)
  - linię taka można uzyskać na Atmega8 (nadawanie 1: PORTx = 0; DDRx = 0; nadawanie 0: PORTx = 0; DDRx = 1;)
- dla każdego slave (1-7) switch ma określoną wartość dopuszczalnego czasu trwania zgody na nadawanie (timeout)

```
for (i = 1; i < 8; ++i) {
        mask = 0x01 << i;
        if (read_oczekujacy_mask() & mask) {
                wykryto_koniec_ramki=0;                               ///
↪   opcjonalnie możemy śledzić koniec ramki i limitować ilość ramek od urządzenia
↪   ....
                start_timer(timeout[i]);
                do {
                        odbieraj_dane(i);                                 ///
↪   transmituj zapytanie
                        while(! (reply = read_oczekujacy_reply_mask()) ); /// czekaj
↪   na zgłoszenie chęci transmisji odpowiedzi
                        odbieraj_dane(reply);                             ///
↪   transmituj odpowiedź
                        while(read_oczekujacy_reply_mask() != 0);         /// czekaj
↪   na koniec transmisji odpowiedzi
                } while (!timer_timeout && !wykryto_koniec_ramki &&
↪   read_oczekujacy_mask() & mask);
                stop_timer(timeout[i]);
        }
}
```

Standard

Aktualnym rozwiązaniem jest zastosowanie szybkiego mastera (podłączanego do innych systemów przy pomocy Ethernetu i sieci TCP/IP) mającego do dyspozycji kilka magistral RS485 i aktywnie odpytującego slave'y z wykorzystaniem standardowego protokołu Modbus[2053] RTU. W wyjątkowych przypadkach możliwe jest także zastosowanie linii przerwaniowej od slave do mastera. Więcej informacji w inteligentny dom.

my_rs485_binary.c

```
/**
 * Copyright (c) 2006-2007, Robert Paciorek (http://www.opcode.eu.org/), BSD/MIT-type license
 *
```

```
 *
 * Redystrybucja wersji źródłowych i wynikowych, po lub bez dokonywania
 * modyfikacji JEST DOZWOLONA, pod warunkiem zachowania niniejszej informacji o
 * prawach autorskich. Autor NIE ponosi JAKIEJKOLWIEK odpowiedzialności za skutki
 * użytkowania tego dokumentu/programu oraz za wykorzystanie zawartych tu informacji.
 *
 * This program is free software. Redistribution and use in source and binary
 * forms, with or without modification, ARE PERMITTED provided save this copyright
 * notice. This document/program is distributed WITHOUT any warranty, use at YOUR
 * own risk.
 *
 **/

/**
 * funkcje obsługugi RS 485
 * zgodne z protokołem komunikacyjnym używanym w moich projektach
 *
 * dwa pierwsze bity (uart_buf_head) oznaczają typ komunikatu:
 *    00 (0x00) - identyfikator urzadzenia
 *    01 (0x40) - identyfikator lini
 *    10 (0x80) - komenda
 *    11 (0xC0) - odpowiedz (ignorowane przez slave)
 */

/** funkcja wysyłająca komendę RS **/
void send_rs_cmd(uint8_t dev, uint8_t line, uint8_t cmd) {
#ifdef DRIVER_CTRL_PIN_SET_DDR
 DRIVER_CTRL_PIN_SET_DDR = DRIVER_CTRL_PIN_SET_DDR | (1 << DRIVER_CTRL_PIN);
#endif
 DRIVER_CTRL_PORT = DRIVER_CTRL_PORT | (1 << DRIVER_CTRL_PIN);
 _delay_ms(5); // 5ms
 UDR=dev;
 _delay_cs(5); // 50ms
 UDR=line;
 _delay_cs(5); // 50ms
 UDR=cmd;
 _delay_ms(5); // 5ms
 DRIVER_CTRL_PORT = DRIVER_CTRL_PORT & ( ~(1 << DRIVER_CTRL_PIN));
#ifdef DRIVER_CTRL_PIN_SET_DDR
 DRIVER_CTRL_PIN_SET_DDR = DRIVER_CTRL_PIN_SET_DDR & ( ~(1 << DRIVER_CTRL_PIN));
#endif
}

/** funkcja wysyłająca odpowiedz przez RS **/
void send_rs(uint8_t reply) {
#ifdef DRIVER_CTRL_PIN_SET_DDR
 DRIVER_CTRL_PIN_SET_DDR = DRIVER_CTRL_PIN_SET_DDR | (1 << DRIVER_CTRL_PIN);
#endif
 DRIVER_CTRL_PORT = DRIVER_CTRL_PORT | (1 << DRIVER_CTRL_PIN);
 _delay_ms(5); // 5ms
 UDR = 0xC0 | reply;
 _delay_ms(5); // 5ms
 DRIVER_CTRL_PORT = DRIVER_CTRL_PORT & ( ~(1 << DRIVER_CTRL_PIN));
#ifdef DRIVER_CTRL_PIN_SET_DDR
 DRIVER_CTRL_PIN_SET_DDR = DRIVER_CTRL_PIN_SET_DDR & ( ~(1 << DRIVER_CTRL_PIN));
#endif
}
```

MY_RS485_ASCII.C

```c
/** plik z funkcjami do obsługi mojego formatu komunikatów RS485
 *    przeznaczony dla mikrokontrolerów AVR
 **/

#ifndef MY_ADDRES
# error "You must define MY_ADDRES"
#endif

/// domyślne ustawienie BAUD (9600 przy 3 MHz, 9600 przy 8 MHz, 300 przy 1 MHz)
#ifndef UBRRL_VAL
# if   F_CPU == 3000000UL
#  define UBRRH_VAL 0
#  define UBRRL_VAL 19
# elif F_CPU == 8000000UL
#  define UBRRH_VAL 0
#  define UBRRL_VAL 51
# elif F_CPU == 1000000UL
#  define UBRRH_VAL 0
#  define UBRRL_VAL 207
#  warning "BAUD = 300 < 9600"
# else
#  error "Nieznane ustawienie UBRR dla zadanego F_CPU"
# endif
#endif

/// dodatkowe opóźnienie po włączeniu nadajnika RS485 (dla half duplex)
// wartości w ms od 0.004 do 32 (przy zegarze pomiędzy 1MHz a 8MHz)
#ifndef RS_NO_WAIT_AFTER_ENABLE
# ifndef RS_WAIT_AFTER_ENABLE
#  define RS_WAIT_AFTER_ENABLE 0.5
# endif
#endif

/// dodatkowe opóźnienie między znakami (gdy odbiorca nie nadąrza z prędkością bitrate)
// wartości w ms od 0.004 do 32 (przy zegarze pomiędzy 1MHz a 8MHz)
#ifndef RS_NO_WAIT_AFTER_CHAR
# ifndef RS_WAIT_AFTER_CHAR
#  define RS_WAIT_AFTER_CHAR 20.0
# endif
#endif

/// obsługa włączania i wyłączania nadajnika RS485
#ifndef RS_EABLE_TX
# if !defined(DRIVER_CTRL_PORT) || !defined(DRIVER_CTRL_PIN)
#  error "You must define RS_EABLE_TX  or  DRIVER_CTRL_PORT and DRIVER_CTRL_PIN"
# endif
# define  RS_EABLE_TX    DRIVER_CTRL_PORT = DRIVER_CTRL_PORT | (1 << DRIVER_CTRL_PIN);
#endif
#ifndef RS_DISABLE_TX
# if !defined(DRIVER_CTRL_PORT) || !defined(DRIVER_CTRL_PIN)
#  error "You must define RS_DISABLE_TX  or  DRIVER_CTRL_PORT and DRIVER_CTRL_PIN"
# endif
# define  RS_DISABLE_TX  DRIVER_CTRL_PORT = DRIVER_CTRL_PORT & ( ~(1 << DRIVER_CTRL_PIN));
#endif

/// obsluga restartowania watchdoga max RESET_WATCHDOG razy
#ifdef RESET_WATCHDOG
 extern uint8_t watchdog_cnt;
```

```
#endif


/** obsługa przerwania związanego z odbiorem danych z portu szeregowego
 *   gdy skompletyujemy cała komendę wywołujemy:
 *      void execute_rs_cmd(uint8_t cmd, uint8_t addr, uint8_t param, uint8_t uart_decode_mode)
 *   która musi być zdefiniowana w programie sterownika danego układu
 *
 *   FORMAT TRANSMISJI:
 *         @adres#subadres#komenda#parametr;
 *      LUB
 *         @adres#subadres#komenda;
 *      GDZIE: adres, subadres, komenda, parametr -> liczba z zakresu 0-255 zapisana ⤸
 dziesiętnie jako ASCII
 *
 *   (ALTERNATYWNY ZNAKOWY) FORMAT TRANSMISJI:
 *         @adres%komendaparametr
 *      GDZIE:
 *         adres -> liczba z zakresu 0-255 zapisana dziesiętnie jako ASCII
 *         komenda, parametr -> pojedyńcze znaki ASCII (każdy poza @)
 **/
void execute_rs_cmd(uint8_t cmd, uint8_t addr, uint8_t param, uint8_t uart_decode_mode);
volatile uint8_t uart_decode_mode;
uint8_t addr_ind, addr, cmd, param;
ISR(USART_RXC_vect) {
 cli();
 uint8_t uart_buf = UDR;

/// dekodowanie komendy
 if (uart_buf == '@') {
  /// poczatek nowej komendy
  uart_decode_mode = 1;
  addr = 0;
  cmd = 0;
  param = 0;
 } else if (uart_decode_mode == 1) {
  /// jest nadawany adres
  if (uart_buf == '#') {
   // koniec adresu
#ifdef MY_MULTICAST_ADDRES
   if (addr == MY_ADDRES || addr == MY_MULTICAST_ADDRES) {
#else
   if (addr == MY_ADDRES) {
#endif
    // bedziemy dekodowac sub-adres
    uart_decode_mode = 2;
   } else {
    // to nie byl nasz adres (ale nie koniec transmisji) wiec czekamy na @ lub ;
    uart_decode_mode = 0xf0;
   }
   addr = 0;
  } else if (uart_buf == '%') {
   // koniec adresu
#ifdef MY_MULTICAST_ADDRES
   if (addr == MY_ADDRES || addr == MY_MULTICAST_ADDRES) {
#else
   if (addr == MY_ADDRES) {
#endif
    // bedziemy dekodowac komendą w postaci znaku alfanumerycznego ASCII
    uart_decode_mode = 10;
```

```
    } else {
      // to nie byl nasz adres (ale nie koniec transmisji) wiec czekamy na @ lub ;
      uart_decode_mode = 0xf0;
    }
    addr = 0;
  } else if (uart_buf == ';') {
    uart_decode_mode = 0; // wykrywanie konca transmisji nawet gdy nie do nas na potrzeby ⤸
    sterowania nadajnikiem
  } else {
    // dekodowanie adresu
    addr = addr * 10 + uart_buf - '0';
  }
} else if (uart_decode_mode == 2) {
  /// jest nadawany sub-adres
  if (uart_buf == '#') {
    // koniec subadresu
    uart_decode_mode = 3;
  } else if (uart_buf == ';') {
    uart_decode_mode = 0; // wykrywanie konca transmisji nawet gdy nie do nas na potrzeby ⤸
    sterowania nadajnikiem
  } else {
    // dekodowanie sub-adresu
    addr = addr * 10 + uart_buf - '0';
  }
} else if (uart_decode_mode == 3) {
  /// jest nadawana komenda
  if (uart_buf == '#') {
    // komenda ma parametr
    uart_decode_mode = 4;
  } else if (uart_buf == ';'){
    // komenda do wykonania
    uart_decode_mode = 0xff;
  } else {
    // dekodowanie komendy
    cmd = cmd * 10 + uart_buf - '0';
  }
} else if (uart_decode_mode == 4) {
  /// jest nadawany parametr
  if (uart_buf == ';') {
    uart_decode_mode = 0xff;
  } else {
    // dekodowanie parametru
    param = param * 10 + uart_buf - '0';
  }
} else if (uart_decode_mode == 10) {
  cmd = uart_buf;
  uart_decode_mode = 11;
} else if (uart_decode_mode == 11) {
  param = uart_buf;
  uart_decode_mode = 0xfe;
} else if (uart_buf == ';') {
  uart_decode_mode = 0; // wykrywanie konca transmisji nawet gdy nie do nas na potrzeby ⤸
  sterowania nadajnikiem
}

/// wykonanie komendy
if (uart_decode_mode > 0xf0) {
  // wykonaj zleconą komendę
  execute_rs_cmd(cmd, addr, param, uart_decode_mode);
  // zaczynamy dekodowanie od poczatku
```

```
  uart_decode_mode = 0;
 }
 sei();
}


/** obsługa inicjalizacji portu szeregowego
 *    funkcja powinna być wywołana w inicjalizującej częśći main()
 **/
void rs_init() {
 // włączenie nadajnika, odbionika UART wraz z generowaniem przerwań
 UCSRB = (1 << TXEN) | (1 << RXEN) | (1 << RXCIE) | (1 << TXCIE);
 // ustawienie prędkości (BAUD), UBRR = F_CPU / (16*BAUD) -1
 UBRRH = UBRRH_VAL;
 UBRRL = UBRRL_VAL;
 uart_decode_mode = 0;
}


/** obsługa włączania i wyłączania nadajnika half-duplex
 *    włącznie przeprowadzane jest jawnym wywołaniem rs_enable()
 *    wyłączanie jest obsługiwane w przerwaniu związanym z końcem transmisji
 *    wymaga aby rs_disable było uprzednio ustawione na 1
 */
volatile uint8_t rs_can_disable;
void rs_enable() {
 // zabraniamy wyłączania w przerwaniu
 rs_can_disable = 0;

// włączamy nadajnik
 RS_EABLE_TX

#ifdef RS_WAIT_AFTER_ENABLE
 // czekamy chwile aż nadajnik będzie gotowy
 _delay_ms( RS_WAIT_AFTER_ENABLE );
#endif
}
void rs_disable() {
 // zezwalamy na wyłączenie w przerwaniu po zakończeniu nadawania
 rs_can_disable = 1;

while ( !( UCSRA & (1<<UDRE)) ); UDR = ';'; // FIXME sztuczka zapewniajaca wylaczenie ⤸
 nadajnika

// jeżeli nadawanie już się zakończyło to od razu wyłączamy
 if (UCSRA & (1 << TXC))
  RS_DISABLE_TX
}
ISR(USART_TXC_vect) {
 // w przerwaniu związanym z zakończeniem transmisji
 // jeżeli mamy zgodę to wyłączamy nadajnik
 if (rs_can_disable == 1) RS_DISABLE_TX
}


/** obsługa wysyłania - znaku oraz liczby 8 bitowej bez znaku
 *    do zewnętrznego użycia
 *       rs_switch_and_send_char()
 *    oraz
 *       rs_switch_and_send_8bit_num()
```

```
 * gdyż tylko one zapewniają właczenie nadajnika
 **/
void rs_send_char(uint8_t byte) {
 // czekamy na miejsce w buforze
 while ( !( UCSRA & (1<<UDRE)) );
 // zlecamy bajt do wysyłania
 UDR = byte;
#ifdef RS_WAIT_AFTER_CHAR
  _delay_ms( RS_WAIT_AFTER_CHAR );
#endif
#ifdef RESET_WATCHDOG
 if (watchdog_cnt < RESET_WATCHDOG) {wdt_reset(); watchdog_cnt++;}
#endif

}

void rs_switch_and_send_char(uint8_t byte) {
 rs_enable();
 rs_send_char(byte);
 rs_disable();
}

void rs_send_8bit_num(uint8_t byte) {
 uint8_t is_send=0;
 uint8_t dzielnik = 100;
 uint8_t cyfra, i;

// wysyłamy kolejne znaki
 for(i=0; i<3; i++) {
  cyfra = (byte/dzielnik)%10;
  dzielnik=dzielnik/10;

  if(is_send != 0 || cyfra != 0 || i == 2) {
   rs_send_char(cyfra + '0');
   is_send=1;
  }
 }
}

void rs_switch_and_send_8bit_num(uint8_t byte) {
 rs_enable();
 rs_send_8bit_num(byte);
 rs_disable();
}

/** obsługa potwierdzania odbioru wiadomości **/
void rs_switch_and_send_ack(uint8_t byte) {
 rs_enable();
 rs_send_char(0x06);
 rs_send_char(byte);
 rs_disable();
}

/** obsługa wysyłania komendy RS
 *    jako że jest to nadawanie inicjalizowane przez nas sprawdzamy zgodę
 *    jednostki zarządzającej magistralą
 **/
#ifdef DRIVER_PERMIT_PORT
int8_t rs_send_cmd(uint8_t addr, uint8_t subadd, uint8_t cmd, uint8_t param, uint8_t is_param,
uint16_t timeout) {
```

```
 while ( timeout && ( DRIVER_PERMIT_PORT & (1 << DRIVER_PERMIT_PIN) ) ) {
  timeout--;
 }
 if ( DRIVER_PERMIT_PORT & (1 << DRIVER_PERMIT_PIN) ) {
  // ndawanie z naszej inicjatywy dozowolone gdy stan niski na PD3
  return -1;
 }
#else
int8_t rs_send_cmd(uint8_t addr, uint8_t subadd, uint8_t cmd, uint8_t param, uint8_t is_param)
{
#endif
 rs_enable();

rs_send_char('@');
 rs_send_8bit_num(addr);
 rs_send_char('#');
 rs_send_8bit_num(subadd);
 rs_send_char('#');
 rs_send_8bit_num(cmd);
 if (is_param) {
  rs_send_char('#');
  rs_send_8bit_num(param);
 }
 rs_send_char(';');

rs_disable();

return 0;
}

#ifdef DRIVER_PERMIT_PORT
int8_t rs_send_char_cmd(uint8_t addr, uint8_t cmd, uint8_t param, uint16_t timeout) {
 while ( timeout && ( DRIVER_PERMIT_PORT & (1 << DRIVER_PERMIT_PIN) ) ) {
  timeout--;
 }
 if ( DRIVER_PERMIT_PORT & (1 << DRIVER_PERMIT_PIN) ) {
  // ndawanie z naszej inicjatywy dozowolone gdy stan niski na PD3
  return -1;
 }
#else
int8_t rs_send_char_cmd(uint8_t addr, uint8_t cmd, uint8_t param) {
#endif
 rs_enable();

rs_send_char('@');
 rs_send_8bit_num(addr);
 rs_send_char('%');
 rs_send_char(cmd);
 rs_send_char(param);
 rs_send_char(';'); // dla automatycznej detekcji konca transmisji

rs_disable();
}


#if 0
ROZWAŻANE KONCEPCJE POPRAWY TEJ METODY TRANSMISJI:
 // PB0
 #define UART_CTRL_PORT  PORTB
 #define UART_CTRL_BIT   0
```

```
void uart_send(char c) {
  uint8_t sreg;

  // wylaczamy przerwania
  sreg = SREG;
  cli();

  // wlaczamy nadajnik
  UART_CTRL_PORT |= 1 << UART_CTRL_BIT;

  // czekamy na miejsce w buforze nadawczym
  while ( !( UCSRA & (1<<UDRE)) );

  // TODO: tu trzeba czy nie trzeba czekac ???

  // zapisujemy do bufora nadawczego
  UDR = c;

  // czyscimy TXC
  UCSRA &= ~(1<<UDRE);

  // przywracamy przerwania
  SREG = sreg;
 }

ISR(USART_TXC_vect) {
  // wylaczamy nadajnik
  UART_CTRL_PORT &= ~(1 << UART_CTRL_BIT);
 }

/*
  #define UART_CTRL_PIN      PINB
  #define UART_CTRL_DDR      DDRB
  void uart_send(char c) {
    cli();
    if (! UART_CTRL_PIN & (1 << UART_CTRL_BIT)) {
      UCSRB = (1 << TXEN) | (1 << RXEN) | (1 << RXCIE) | (1 << TXCIE);
      UART_CTRL_PORT |= 1 << UART_CTRL_BIT;
      _delay_ms( RS_WAIT_AFTER_ENABLE ); // to naprawdę potrzebne ???
      UDR = c;
    }
    sei();
  }

  void uart_send(char c) {
    if (! (UCSRB & (1 << TXEN)) ) {
      UART_CTRL_PORT |= 1 << UART_CTRL_BIT;
      UCSRB |= (1 << TXEN) | (1 << TXCIE);
    }
    UDR = c;
  }
  ISR(USART_TXC_vect) {
    // disable transmiter on AVR
    UCSRB &= ~((1 << TXEN) | (1 << TXCIE));
    // disable external transmiter
    UART_CTRL_PORT &= ~(1 << UART_CTRL_BIT);
  }
 */
```

```
#endif
```

STEROWNIK_OSWIETLENIA_V3.1.C

```
/*
 * program sterownika oświetlenia z funkcją zdalnego sterowania poprzez port RS
 * dla mikrokontrolera Atmega8
 *
 *
 * kompilacja i wgranie przez ISP:
 *    avr-gcc -mmcu=atmega8 -Os -o sterownik.o sterownik_oswietlenia.c
 *    avr-objcopy -O ihex sterownik.o sterownik.hex
 *    uisp -dprog=dapa -dlpt=0x378 --erase --upload --verify --segment=flash if=sterownik.hex
 *
 *    w razie problemow warto przeladowac modul odpowiedzialny za obsluge LPT
 *        rmmod lp parport_pc parport ; modprobe parport_pc
 *
 * konfiguracja układu na zewnętrzny kwarc 8MHz (lub troche mniejszy)
 *    uisp -dprog=dapa -dlpt=0x378 --wr_fuse_l=0xff
 */

#include <inttypes.h>
#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>

/** nagłówki funkcji opóżniających **/
#define F_CPU 8000000UL
#include <util/delay.h>

/** nagłówki mojego formatu transmisji przez RS485 **/
#define DRIVER_CTRL_PORT PORTD
#define DRIVER_CTRL_PIN PD4
#define MY_ADDRES 11

#include "my_rs485_ascii.c"


/** ustawienia sciemniacza **/
/// ilość linii obsługujących ściemnianie
///
/// wymagamy aby linie z regulacją PWM (ściemniaczem) stanowiły ciągły zakres od ⤸
początku
/// przyjętej kolejności pinów (0...14): PD5,...,PD7,PB0,...,PB5,PC0,...,PC5
/// do linii ściemniających powinien być podłączony MOC3023 a nie MOC3043
///
/// wartość zerowa REG_NUMB powoduje przełączanie wyjścia bezpośrednio w obsłudze ⤸
wykonywania komendy RS
/// procedura obsługi przerwania związanego z detekcją zera nie jest instalowana
/// umożliwia to pominięcie w takim wypadku układu detekcji zera
#define REG_NUMB 15

/// maksymalna ilość poziomów regulacji ściemnienia
#define COUNT_MAX 0xdf

/// opóżnienienie obiegu pętli w us
///
/// wartość należy dobrać tak aby czas wykonywania procedury obsługi przerwania był ⤸
```

```
około 75% okresu pomiędzy przerwaniami
/// częstotliwosc_przerwań = 2 * czastotliwosc_sieci (sygnał związany z przejściami ⤸
przez zero)
/// UWAGA: na czas wykonywania procedury obsługi przwerwania wpływa także: F_CPU, REG_NUMB,⤸
COUNT_MAX
#define COUNT_OFFSET 0.2

/// opóźnienie rozpoczęcia pętli sterowania ścienieniem
/// związane z wywoływaniem na zboczu narastającym
/// wartość należy zmierzyć eksperymentalnie
#define START_OFFSET 0.17

/// konfiguracja wyjsc - makro odpwiedzialne za przepisywanie ports na piny uC
///
/// jeżeli pod którejś z pinów podłączone są rejestry sterujące przekaźnikami
/// lub optotriakami z detekcją zera wyjścia te należy wyłączyć z obsługi ports,
/// podobnie należy uczynić z pinami wykorzystywanymi jako dane dla tych zatrzasków
///
/// piny te powinny być obsługiwane bezpośrednio w "switch (cmd)" w USART_RXC_vect
/// [adres rejestru ustalacj jako (addr-LICZBA_STEROWANYCH_BEZPOSREDNIO)⤸
%LICZBA_BITOW_REJESTROW]
/// jako iż wprowadzenie takiego trybu wiązałoby się z możliwością przepełnienia ⤸
zmiennej status
/// należy rozważyć użycie kilku zmiennych o takim przeznaczeniu (np. tablicy uint8_t ⤸
indeksowanej numerem rejestru)
/// lub rezygnacje z funkcji pamięciowych (read/switch)
#define SET_OUTPUT(val) PORTD=(val & 0xf) << 5; PORTB=(val >> 3) & 0x3f; PORTC=(val >> 9) & ⤸
0x3f;

/** tryb debugowania (nie używać w sieci RS) **/
//#define DEBUG
//#define DEBUG_TIME

#ifdef DEBUG
# warning "Enable DEBUG MODE - don't use in RS network"
#endif

#ifdef DEBUG_TIME
volatile double zero_offset = START_OFFSET;
#endif


/** konfiguracja "czasu pracy" **/
// wykorzystywane w poniższych deklaracjach volatile pozwala na
// korzystanie z tej samej zmiennej w przerwaniach i w main()
// umożliwia uniknięcie stosowania sztuczek wskaznikowych typu:
//    double * opoznienie_wsk = &opoznienie;
//    opoznienie = *opoznienie_wsk;

// maska bitowa przechowująca informację o stanie linii
volatile uint16_t status;

// opoznienie sciemnienia
volatile uint8_t opoznienie[REG_NUMB];


/** obsluga portu szeregowego **/
void execute_rs_cmd(uint8_t cmd, uint8_t addr, uint8_t param, uint8_t uart_decode_mode) {
 addr--; // chcemy aby linie były numerowa zgodnie z elementami na PCB czyli od 1
 switch (cmd) {
```

```
  case 0x01: /// read
   if ( (status & (1 << addr)) != 0 )
    rs_switch_and_send_char('1'); // ON
   else
    rs_switch_and_send_char('0'); // OFF
   break;
  case 0x02: /// on
   status = status | (1 << addr);
#if REG_NUMB == 0
   SET_OUTPUT(status);
#endif
   break;
  case 0x03: /// off
   status = status & ~(1 << addr);
#if REG_NUMB == 0
   SET_OUTPUT(status);
#endif
   break;
  case 0x04: /// switch
   status = status ^ (1 << addr);
#if REG_NUMB == 0
   SET_OUTPUT(status);
#endif
   break;

  case 0x05: /// odczyt wartosci sciemnienia lub ststusu
   if ( addr < REG_NUMB) { // sciemnienie zadanej linii
    cmd = opoznienie[addr];
   } else if (addr == REG_NUMB) { // młodszy bajt statusu
    cmd = status;
   } else if (addr == REG_NUMB+1) { // starszy bajt statusu
    cmd = status >> 8;
   } else {
    break;
   }
   rs_switch_and_send_8bit_num( opoznienie[addr] );
   break;
  case 0x06: /// ustawienie wartosci sciemnienia lub ststusu
   if ( addr < REG_NUMB) { // sciemnienie zadanej linii
    opoznienie[addr] = param;
   } else if (addr == REG_NUMB) { // młodszy bajt statusu
    status = (status & 0xff00) | param;
   } else if (addr == REG_NUMB+1) { // starszy bajt statusu
    status = (status & 0x00ff) | param;
   }
   break;
#ifdef DEBUG_TIME
  case 10:
   zero_offset=addr/100.0;
   break;
  case 12:
   MCUCR = (1 << ISC00) | (1 << ISC01); // narastajace
   break;
  case 13:
   MCUCR = 0; // niski
   break;
  case 14:
   MCUCR = (1 << ISC00); // zmiana stanu
   break;
  case 15:
```

```
   MCUCR = (1 << ISC01); // opadajace
   break;
#endif
 }
}


#if REG_NUMB != 0
/** obsluga przerwania powodowanego przejsciem przez zero fazowe **/
// w razie problemów z działaniem przerwania zegarowego mozna sprobowac
// to wrzucic do petli glownej obarczajac stosownym if'em na zmiane stanu PD2:
// new=PIND & (1 << PD2); if (new != last) {last=new; ...}
uint8_t i, j;
uint16_t ports;
ISR(INT0_vect) {
 cli();

ports = (0xffff << REG_NUMB) & status;
 // jako wstępną wartość przyjmujemy stan linii nie będących ściamniaczami zgodny z ⤸
 statusem

#ifdef DEBUG_TIME
 _delay_ms(zero_offset);
#else
 _delay_ms(START_OFFSET);
#endif

/// aktywujemy linie ściemniacza z stosownymi opóżnieniami
 for (i=0; i<COUNT_MAX; i++) {
  for (j=0; j<REG_NUMB; j++) {
   if (opoznienie[j] == i) {
    // włączamy triak odpowiedzialny za linię j
    // pod warunkiem że linia aktywna
    ports = ports | (status & (1 << j));
   }
  }

  SET_OUTPUT(ports);
  // powyzsze instrukcje moglyby być w wewnetrznym if:
  // wtedy wykonywaly by sie duzo rzadziej,
  // ale moglyby zaburzac rownomiernosc opoznienia

  _delay_us(COUNT_OFFSET);
 }

/// wyłączenie linii dla których ściemnienie !=0
 for (j=0; j<REG_NUMB; j++) {
  if (opoznienie[j] != 0) {
   // wyłczamy triak odpowiedzialny za linię j
   // pod warunkiem że linia ma niepełną jasność
   ports = ports & (~(1 << j));
  }
 }
 SET_OUTPUT(ports);

sei();
}
#endif
```

```
main() {
 /** konfiguracja portów IO mikrokontrolera **/
// port B - wyjściowy
 DDRB = 0xff;
 PORTB = 0;

// port C - wyjściowy
 DDRC = 0xff;
 PORTC = 0;

// port D - wyjściowo/ wejściowy:
 //    PD2 wejsciowy bez podciagajacego
 //    PD4-PD7 wyjściowe
 DDRD = (1 << PD4) | (1 << PD5) | (1 << PD6) | (1 << PD7);
 PORTD = 0x00;

/** port szeregowy - inicjalizacja **/
rs_init();

#ifdef DEBUG
 UDR = 'S'; _delay_ms(1000);
 UDR = 'O'; _delay_ms(1000);
 UDR = ' '; _delay_ms(1000);
 UDR = 'v'; _delay_ms(1000);
 UDR = '2'; _delay_ms(1000);
 UDR = '.'; _delay_ms(1000);
 UDR = '9'; _delay_ms(1000);
 UDR = '.'; _delay_ms(1000);
 UDR = '1'; _delay_ms(1000);
#endif

#if REG_NUMB != 0
 /** przerwania z int0 **/
GIMSK = (1 << INT0);
 MCUCR = (1 << ISC00) | (1 << ISC01);
 // reagujemy na zbocze narastajace, a nie opadajace (1 << ISC01) i czekamy chwile w ⤸
 procedurze obslugi przerwania
 // podyktowane jest to minimalizacją czasu potrzebnego na włączenie gdy ściemnienie = 0
#endif
 /** wartość początkowe zmiennych **/
status = 0x0000;
 for (j=0; j<REG_NUMB; j++)
  opoznienie[j] = 0;

/** włączenie przerwań **/
sei();

/** petla glowna **/
while (1) {
 }
}
```

STEROWNIK_OSWIETLENIA_V2.0.C

```
/**
 * Copyright (c) 2005-2007, Robert Paciorek (http://www.opcode.eu.org/), BSD/MIT-type license
 *
 *
```

```c
 * Redystrybucja wersji źródłowych i wynikowych, po lub bez dokonywania
 * modyfikacji JEST DOZWOLONA, pod warunkiem zachowania niniejszej informacji o
 * prawach autorskich. Autor NIE ponosi JAKIEJKOLWIEK odpowiedzialności za skutki
 * użytkowania tego dokumentu/programu oraz za wykorzystanie zawartych tu informacji.
 *
 * This program is free software. Redistribution and use in source and binary
 * forms, with or without modification, ARE PERMITTED provided save this copyright
 * notice. This document/program is distributed WITHOUT any warranty, use at YOUR
 * own risk.
 *
**/

/*
 * program sterownika oświetlenia z funkcją zdalnego sterowania poprzez port RS
 * dla mikrokontrolera Atmega8
 *
 *
 * kompilacja i wgranie przez ISP:
 *    avr-gcc -mmcu=atmega8 -Os -o sterownik.o sterownik_oswietlenia.c
 *    avr-objcopy -O ihex sterownik.o sterownik.hex
 *    uisp -dprog=dapa -dlpt=0x378 --erase --upload --verify --segment=flash if=sterownik.hex
 *    w razie problemow warto przeladowac modul odpowiedzialny za obsluge LPT
 *       rmmod lp parport_pc parport ; modprobe parport_pc
 *
 * konfiguracja układu na zewnętrzny kwarc 3MHz
 *    uisp -dprog=dapa -dlpt=0x378 --wr_fuse_l=0xEF
 */


#include <inttypes.h>

#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>

/** nagłówki funkcji opóźniających **/
#define F_CPU 3000000UL
#include "opoznienia.c"

/** nagłówki mojego formatu transmisji przez RS485 **/
#define DRIVER_CTRL_PORT PORTD
#define DRIVER_CTRL_PIN PD7
//#define DRIVER_CTRL_PIN_SET_DDR DDRD
#include "my_rs485.c"

#define ZMIANA_OPOZNIENIA 0.05

/// identyfikator urządzenia liczba z zakresu 0 - 63
#define _DEV_ID_ 0x01

/// komendy RS wysyłane przez guziki specjalne
#define PD6_RS_CMD send_rs_cmd(0x01, 0x46, 0x82)

/// tryb debugowania (nie używać w sieci RS)
//#define DEBUG 1

#ifdef DEBUG
# warning "Enable DEBUG MODE - don't use in RS network"
# define DEBUG_SEND_OPOZNIENIE (uint8_t)(opoznienie/ZMIANA_OPOZNIENIA)
uint8_t kontr=0;
```

```
#endif

/// maska bitow przechowuąca informacje o zablokowanych liniach przełącznika
volatile uint16_t block_mask = 0;

/// maska bitowa przechowująca informację o stanie linii
volatile uint8_t line_status=0x00;

/// opoznienie sciemnienia
volatile double opoznienie=0.00;
volatile uint8_t opoznienie_ctrl=0;

// wykorzystywane w powyższych deklaracjach volatile pozwala na
// korzystanie z tej samej zmiennej w przerwaniach i w main()
// umożliwia uniknięcie stosowania sztuczek wskaznikowych typu:
//    double * opoznienie_wsk = &opoznienie;
//    opoznienie = *opoznienie_wsk;

/** obsluga portu szeregowego **/
uint8_t uart_buf_head, uart_buf_body, chanel, cmd = 0;
ISR(USART_RXC_vect) {
 cli();

uart_buf_head = UDR;
#ifdef DEBUG
 UDR = uart_buf_head; // echo
#endif
 uart_buf_body = uart_buf_head & 0x3F;
 uart_buf_head = uart_buf_head & 0xC0;
 /// dwa pierwsze bity (uart_buf_head) oznaczają typ komunikatu:
 /// 00 (0x00) - identyfikator urzadzenia
 /// 01 (0x40) - identyfikator lini
 /// 10 (0x80) - komenda
 /// 11 (0xC0) - odpowiedz (ignorowane przez slave)

/// kolejnosc nadawania pelnej komendy id urzadzania, id lini, komenda

if (uart_buf_head != 0x00 && cmd != 0) {
  /// gdy nie ID urządzenia oraz wcześniejszy ID urządzenia pasował do naszego
  if (uart_buf_head == 0x40) {
   chanel = uart_buf_body;
   cmd = 2; /// przed komendą wymagany jest identyfikator linii
  } else if (uart_buf_head == 0x80 && cmd == 2) {
   switch (uart_buf_body) {
    case 0x01: /// read
     if ( (line_status & (1 << (chanel - 1))) != 0 )
      send_rs(0x01); /// YES
     else
      send_rs(0x02); /// NO
     break;
    case 0x02: /// on
     line_status = line_status | (1 << (chanel - 1));
     break;
    case 0x03: /// off
     line_status = line_status & ~(1 << (chanel - 1));
     break;

    case 0x04: /// block read
     if ( (block_mask & (1 << (chanel - 1))) != 0 )
```

```
        send_rs(0x01); /// YES
       else
        send_rs(0x02); /// NO
       break;
     case 0x05: /// block on
       // chanel == 7,8   => linie specjalne (PD6, PD3)
       // chanel == 9,10 => ściemnienie (up, down)
       block_mask = block_mask | (1 << (chanel - 1));
       break;
     case 0x06: /// block off
       block_mask = block_mask & ( ~ (1 << (chanel - 1)) );
       break;

     case 0x07: /// ustawienie wartosci sciemnienia
       opoznienie = chanel * ZMIANA_OPOZNIENIA;
       opoznienie_ctrl = chanel;
       break;
     case 0x08: /// odczyt wartosci sciemnienia
       send_rs( opoznienie_ctrl );
       break;

     case 0x09: /// zmiana stanu linii
       line_status = line_status ^ (1 << (chanel - 1));
       break;
#ifdef DEBUG
     case 0x0a:
       UDR=PIND;
       break;
     case 0x0b:
       MCUCR = (1 << ISC00) | (1 << ISC01); // narastajace
       opoznienie=0.00; opoznienie_ctrl = 0;
       break;
     case 0x0c:
       MCUCR = 0; // niski
       opoznienie=0.00; opoznienie_ctrl = 0;
       break;
     case 0x0d:
       MCUCR = (1 << ISC00); // zmiana stanu
       opoznienie=0.00; opoznienie_ctrl = 0;
       break;
     case 0x0e:
       MCUCR = (1 << ISC01); // opadajace
       opoznienie=0.00; opoznienie_ctrl = 0;
       break;
#endif
    }
    cmd = 0; /// po wykonaniu komendy oczekujemy na całą nowa paczkę
  }
 } else if (uart_buf_body == _DEV_ID_) {
  /// gdy ID naszego urządzenia
  cmd = 1;
 } else {
  /// gdy ID innego urządzenia
  cmd = 0;
 }
 sei();
}

/** obsluga przerwania powodowanego przejsciem przez zero fazowe **/
SIGNAL (SIG_INTERRUPT0) {
```

```
 _delay_ms(0.00 + opoznienie); // MAX 0x3f * ZMIANA_OPOZNIENIA < 4ms
 PORTB = line_status;
 _delay_us(10); // 10us

if (opoznienie_ctrl != 0)
  PORTB = line_status & 0xdf; // wyłaczamy PIN6

#ifdef DEBUG
 //if (kontr++ == 100) {kontr=0; send_rs(DEBUG_SEND_OPOZNIENIE);}
#endif
}


main() {
 /** konfiguracja portów IO mikrokontrolera **/
/// port B - wyjściowy
 DDRB = 0xff;
 PORTB = 0;

/// port C - wejściowy z rezystorami podciągającymi
 DDRC = 0;
 PORTC = 0xff;

/// port D - wyjściowo/ wejściowy:
 ///   PD2 wejsciowy bez podciagajacego
 ///   PD3 wejsciowy z podciagajacym
 ///   PD4-PD5 wyjściowe/wejsciowe z podciagajacym
 ///   PD6 wejsciowy z podciagajacym
 ///   PD7 - wyjsciowy, wyłączony
 DDRD = (1 << PD4) | (1 << PD5) | (1 << PD7);
 PORTD = (1 << PD3) | (1 << PD4) | (1 << PD5) | (1 << PD6);

/** port szeregowy - inicjalizacja **/
/// włączenie nadajnika, odbionika UART wraz z generowaniem przerwań
 UCSRB = (1 << TXEN) | (1 << RXEN) | (1 << RXCIE);
 /// ustawienie prędkości (BAUD), UBRR = F_CPU / (16*BAUD) -1
#if F_CPU == 3000000UL
 UBRRH = 0; UBRRL = 19; // BAUD = 9600 przy 3 MHz
#elif F_CPU == 1000000UL
 UBRRH = 0; UBRRL = 207; // BAUD = 300 przy 1 MHz
# warning "BAUD = 300 < 9600"
#else
# error "Nieznane ustawienie UBRR dla zadanego F_CPU"
#endif

/** przerwania z int0 **/
GIMSK = (1 << INT0);
 MCUCR = (1 << ISC01);

/** włączenie przerwań **/
sei();

/** petla glowna **/
uint8_t i, port_in, last=0x00;
 while (1) {
  /// sterowanie ściemnieniem
  port_in = PIND;

  // świecenie LED na liniach ściemniania + na razie wygaszamy
  DDRD = (1 << PD4) | (1 << PD5) | (1 << PD7);
```

```
PORTD = (1 << PD3) | (1 << PD6);

//gdy możemy zmniejszać
if ( (opoznienie_ctrl > 0) && (0 == (block_mask & (1 << 8))) ) {
 // gdy polecono zmniejszanie
 if ( ((port_in & 0x10) == 0) ) {
  opoznienie_ctrl--;
  opoznienie = opoznienie_ctrl * ZMIANA_OPOZNIENIA;
  #ifdef DEBUG_SEND_OPOZNIENIE
  UDR=DEBUG_SEND_OPOZNIENIE;
  #endif
 }
 PORTD = PORTD | (1 << PD4);
}

// gdy możemy zwiększać
if ( (opoznienie_ctrl < 0x3f) && (0 == (block_mask & (1 << 9))) ) {
 // gdy polecono zwiększanie
 if ( (port_in & 0x20) == 0 ) {
  opoznienie_ctrl++;
  opoznienie = opoznienie_ctrl * ZMIANA_OPOZNIENIA;
  #ifdef DEBUG_SEND_OPOZNIENIE
  UDR=DEBUG_SEND_OPOZNIENIE;
  #endif
 }
 PORTD = PORTD | (1 << PD5);
}


/// sterowanie on-off
port_in = (PINC & 0x3f) | (port_in & 0x40) | ((port_in << 4) & 0x80); // zapamietujemy stan ⤸
portu
for (i=0; i<6; i++) {
 if ( 0 != (block_mask & (1 << i)) )
  continue;

 if ( (0 == (port_in & (1 << i))) && (0 != (last & (1 << i))) ) {
  // jest stan niski a poprzednio byl wysoki wiec zmieniamy stan wyjscia
  line_status = line_status ^ (1 << i);
  #ifdef DEBUG
  UDR=line_status;
  #endif
 }
}


/// wysyłanie komend RS
if ( (0 == (port_in & (1 << 6))) && (0 != (last & (1 << 6))) && (0 == (block_mask & (1 << 6)⤸
)) ) {
 PD6_RS_CMD;
}
/* nie działa poprawnie :-( ... konflikt z int0 :-s
if ( (0 == (port_in & (1 << 7))) && (0 != (last & (1 << 7))) && (0 == (block_mask & (1 << 7)⤸
)) ) {
 PD3_RS_CMD;
}*/
last = port_in; // zapamietujemy poprzedni stan portu


/// zapobieganie zawieszeniom ...
```

```
  PORTC = 0xff;

  /// zapobieganie szumom z przelacznikow oraz opoznienie
  /// dla regulacji ciągłej ściemnienia i sygnalizacji LED
  _delay_cs(5);

  /// musi być możliwość odczytu linii ściemniania - koniec świecenia, ustawienie ⤸
  podciągających
  DDRD = (1 << PD7);
  PORTD =  (1 << PD3) | (1 << PD4) | (1 << PD5) | (1 << PD6);
  _delay_us(10);
 }
}
```

Programy dla elektronika, elektryka itp.

Na stronie tej zamieszczam informacje dotyczące korzystania z wolnych programów przydatnych w elektronice, elektryce itp. Ich (zapewne niepełną) listę znalężść moża w spisie oprogramowania

Oprogramowanie inżynierskie

- **oprogramowanie do projektowania układów elektronicznych**
  - gEDA[2054] - projektowanie układów elektronicznych (w szczególności **gschem**), @ Wikipedia[2055]
  - PCB[2056] - projektowanie płytek układów elektronicznych
  - xgsch2pcb[2057] - program ułatwiający równoczesną pracę w gsch i pcb
  - KiCAD[2058] - alternatywny dla gEDA + PCB czyli prgram do projektowania układów elektronicznych oraz płytek dla nich, ponadto ładnie razdzi sobie z obwodami elektrycznymi (patrz `/usr/share/doc/kicad/demos/electric`)
  - Liquid PCB[2059] - orginalne podejście do projektowania PCB
  - xcircuit[2060] - tworzenie schematów
- **symulatory układów elektronicznych**
  - ktechlab[2061] - IDE dla elektroniki i mikrokontrolerów dla KDE, program fajnie symuluje układy elektroniczne
  - oregano[2062] - frondend do gnucap / ngspice
  - gnucap[2063] - program do symulacji układów elektronicznych
  - ngspice[2064] - kontynuacja znanegi i słynnego programu do symulacji układów elektronicznych
  - klogic[2065] - program do projektowania i testowania schematów logicznych
  - ksimus[2066] -
  - qucs[2067] -
  - tkgate[2068] -
- **programatory**
  - uisp[2069] - programator ISP
  - avrdude[2070] - programator ISP i JTAG dla AVR
  - AVaRICE[2071] - debuger/programator JTAG
  - avrprog[2072] - prosty programator dla AVR
  - avrp[2073] - prosty programator dla AVR
- **HEX-edytory**
  - - edytor szesnastkowy z KDE
  - - edytor szesnastkowy z GNOME
- **Programy do obsługi portu szeregowego**
  - minicom[2074] - terminal portu szeregowego
  - picocom[2075] - prosty emulator terminala dla portów szeregowych pracujhący w konsoli (zakończenie przez Ctrl+A + Ctrl+X)
  - jpnevulator[2076] - konsolowy słuchacz portu szeregowego, z opcją nadawania (potrafi wyświrtlać i wywsyłać dane szesnastkowe)
  - gtkterm[2077] - prosty emulator terminala dla portów szeregowych pracujący w środowisku graficznym (umożliwia wysyłanie i odbieranie danych jako dane zapisane szesnastkowo)

    - cutecom[2078] - kolejny prosty emulator terminala dla portów szeregowych pracujący w środowisku graficznym umożliwiający wysyłanie i odbieranie danych jako dane zapisane szesnastkowo
- **Programy do tworzenia schematów, diagramów oraz programy CAD**
    - librecad[2079] - fork QCad'a, @ Wikipedia[2080]
    - qcad[2081] - CAD, @ Wikipedia[2082]
    - freecad[2083] - CAD 3D
    - pythoncad[2084] - CAD
    - OpenSCAD[2085] - bazujący na edycji skryptowej CAD 3D
    - dia[2086] - edytor diagramów
    - archimedes[2087] - CAD
    - graphviz[2088] - system wizualizacji grafów
- gerbv[2089] - przeglądarka plików Gerber
- ghdl[2090] - symulator VHDL
- gtkwave[2091] - prezenter wykresów symulacyjnych (np. z ghdl)
- Icarus Verilog[2092] - narzędzie symulacji i syntezy Verilog
- espresso[2093] - minimalizacja funkcji logicznych, zamieszam także skrypt do generowania plików PLA na podstawie wyrażenia boolowskiego - expr2pla.sh.
- baudline[2094] - wizualizacja spectrum

Środowiska symulacyjne

Do najpopularniejszych środowisk symulacji układów elektronicznych należy SPCICE. Niejako jego klonem jest GnuCAP, które w dużym stopniu jest zgodny na poziomie składni plików opisujących układ i symulację, jednak występują tu pewne rozbieżności. Zamieszczam demonstracyjny plik z opisem symulacji[2095] zgodny z obydwoma tymi symulatorami.

Środowiska symulacyjne/obliczeniowe są używane także w technice cyfrowej. Zamieszam także skrypt do plików PLA (z liczbą wyjść równą 1) dla podanej formuły boolowskiej - expr2pla.sh

Innym, aczkolwiek dopiero raczkującym (za to dość przyjemnym) środowiskiem jest Ktechlab, tutaj także zamieszczam przykładowy plik[2096] dla tego środowiska.

expr2pla.sh

```
#!/bin/bash

# skrypt generuje plik PLA (z liczbą wyjść równą 1) dla podanej formuły boolowskiej
# formuła ta jest wyrażeniem zapisanym przy pomocy operatorów logiki bitowej:
#       a AND b    => a & b
#       a OR b     => a | b
#       a XOR b    => xor(a,b)
#       NOT a      => not(a)
#    argumenty można grupować z wykorzystaniem nawiasów ()
#    kolejne argumenty nazywane są x0 x1 ... xN
#    np. "x1 & ( x2 | x0 )" lub "( not(x0 & x1) | x2 ) & x3"

# pliki PLA są wykorzystywane przez systemy minimalizacji takie jak espresso

FORMULA=$1

MAX_X=`echo "$FORMULA" | awk 'BEGIN{FS="[ |&,)(]"; RS="x"; MAX=0} NR>1{if (MAX<$1) MAX=$1} 
END{print MAX}'`

# definujemy dla calc bitowy not nie dokonujący przeniesienia dla liczb <= 32 bitowych
CALC="define not(a) {return (0xFFFFFFFF >> (32-digits(a,2))) & ~a};"

# genrujemy skrypt dla calc
CALC_1=""
CALC_2="printf(\""
CALC_3=""
```

```
for i in `seq 0 $MAX_X`; do
 CALC_1="$CALC_1 for (x$i=0; x$i<=1; x$i++)"
CALC_2="$CALC_2%b"
CALC_3="$CALC_3 x$i, "
done
CALC_2="$CALC_2 %b\n\", $CALC_3 $FORMULA );"

echo .type fr
echo .i $(( $MAX_X+1 ))
echo .o 1
echo .p `calc "2^($MAX_X+1)" | grep -v defined`

calc "$CALC $CALC_1 $CALC_2"

echo .end
```

GEDA I PCB

Jest to zestaw oprogramowania służącego do tworzenia schematów elektronicznych (gschem) oraz projektów płytek drukowanych (pcb).

Porady użytkowe

Na wstępie kilka użytecznych skrótów klawiaturowych w PCB:

- INFORMACYJNE, POMOCNICZE:
  - Shift+D - pokazanie numeracji pinów w suboknie
  - D - pokazanie numeracji pinów (lub numeru wskazanego pinu) na schemacie
  - F - wyszukanie połączeń
  - O - aktualizacja "rat line"
- ZMIANY ROZMIARÓW ODSTĘPÓW, ITP.:
  - S / Shift+S - zmiana rozmiaru obwódki wskazanego elementu (np. grubości linii rysunku warstwy opisowej lub grubości pola lutowniczego wokół)
  - Alt+S / Alt+Shift+S - zmiana rozmiaru otworu wskazanego elementu (pinu, via, ...)
  - Q - zmiana kształtu (koło/kwadrat) wskazanego pinu lub wszystkich pinów w wskazanym elemencie
  - K / Shif+K - zmiana odstępu pomiędzy linią, via, pinem, etc a wielobokiem
  - J - przełączenie odstępu pomiędzy linią a wielobokiem
- INNE:
  - B / Shift+B - przeniesienie wskazanego / zaznaczonych elementów na drugą stronę płytki (nie działa dla lini, w tym tekstu - można poprawić ręcznie w pliku poprzez dodanie "onsolder" w atrybutach tekstu)
  - H - ukrycie nazw
  - zmiana narzędzia: F2 - rysowanie linii, ..., F9 - obrót elementu

W gschem występujądwa atrybuty o podobnych nazwach - "netname" oraz "net". Warto wiedzieć że pierwszy z nich służy do nazywania danego drutu i umożliwia łączenie poprzez nazwy czy też tworzenie magistral (bus). Drugi jest postaci "nazwa:pin" i służy do powiedzenia że do danego pinu podłączony (wirtualny) drucik o wskazanej nazwie. Umożliwia to nadpisanie domyślnych podłączeń w układach (gnd, vcc, itp) a także stworzenie jedno-pinowego elementu który będzie podłączał się do wskazanej w atrybucie net sieci.

Współpraca między gschem i pcb - gsch2pcb

Istnieje możliwość wygenerowania na podstawie schematu programu gschem (z pakietu gEDA) materiału wejściowego dla projektowania płytki w pcb. Aby to zrobić w trakcie edycji schematu (lub po niej) zainteresować powinniśmy się atrybutem "footprint" odpowiedzialnym za przechowywanie informacji jaki element PCB odpowiada temu symbolowi. Poniżej prezentuję listę najistotniejszych footprint'ów.

Gdy skończymy uzupełnianie naszego schematu powinniśmy wygenerować na jego podstawie plik *.pcb oraz *.net - posłuży do tego program gsch2pcb. Później otwieramy nasz plik w programie pcb i ładujemy odpowiedni plik *.net (File -> Load netlist file). Następnie możemy już układać naszą płytkę (na poczatku warto skorzystać z

opcji która rozsunie elementy po płytce - wybieramy Select -> Disperse all elements), aby odświeżyć mapę połączeń wciskamy O (lub wybieramy Connects -> Optimze rasts-net).

W przypadku gdy zaktualizowaliśmy schemat i chcemy także poprawić projekt płytki należy ponownie wywołać (na tym samym pliku) `gsch2pcb` i postąpić zgodnie z wypisanymi przez niego instrukcjami czyli wkleić nowododane elementy oraz zaktualizować mapę połączeń. Warto pamiętać także o możliwości wygenerowania tylko odświeżonej listy połączeń (netlist) przy pomocy `gnetlist -g PCB plik.sch`, a także o `xgsch2pcb` automatyzującym opisany tu proces (wygodne przy równoczesnej pracy nad schematem i pcb).

Jeżeli kożystamy z footprintów przechowywanych w `~/.pcb/elements/` warto do `~/.bashrc` dodać:

```
function gsch2pcb() {
        command gsch2pcb $(for f in ~/.pcb/elements/*; do echo -d $f; done) "$@"
}
```

Przy drukowaniu (eksporcie) projektów z pcb warto zwrócić uwagę na ustawienia grup warstw (menu "file - preferences", zakładka "layers - group"), można również wyłączyć niepotrzeebne nam elementy wydruku w menu wydruku/eksportu.

Hierarchiczne schematy

W gschem możliwe jest wstawianie pod-schematów jako elementów na schemacie. Ułatwia to projektowanie dużych i powtarzalnych układów. Wymaga to dodania do pliku konfiguracyjnego `~/.gEDA/commonrc` wpisu:

```
(component-library ".")
(source-library  ".")
```

W schemacie podukładu umieszczamy elementy "in-1" lub "out-1" z odpowiednio ustawionym atrybutem `rdefs` (będzie to nazwa pinu symbolu wstawionego do głównego schematu. Następnie tworzymy (niestety trzeba to zrobić ręcznie) symbol takiego komponentu, pamiętając o umieszczeniu atrybutu `source` wskazującego na plik zawierający schemat podukładu oraz umieszczeniu pinów z odpowiednim atrybutem `pinlabel` (zgodnym z `rdefs` elementów "in-1" lub "out-1" w schemacie). Tak przygotowany symbol możemy wykorzystać w schemacie głównym, a używając menu kontekstowego na nim możemy przechodzić do edycji pod-schematu. Zamieszczam skrypt ułatwiający generowanie PCB głównego układu w oparciu o PCB zaprojektowane dla pod-schematów: gsch2pcb-gen_sub_pcb.sh.

gsch2pcb-gen_sub_pcb.sh

```
#!/bin/bash

# skrypt służy do generacji plików pcb dla podschematów
# generacja polega na zmienieniu oznaczen elementów na bazowym pcb podschematu
# tak aby byly one zgodne z autoamtycznie generowana nettlista
# (gdzie pozimoy w oznaczeniach elementow odzielane sa poprzez /)
#
# tak uzyskane schematy włącza sie do glownego pcb co przy powtarzalnych strukturach
# zaoszczędza sporo pracy z układaniem po kilka razy takich samych fragmentów



# generowanie plików konfiguracyjnych gEDA
echo -n "Generate gEDA rcfiles ... "

echo '; zapewniamy ze source jest poszukiwane w katralogu bierzacym' > gafrc
echo '(source-library  ".")' >> gafrc
echo '; zapewniamy ze symbole sa poszukiwane w katralogu bierzacym' >> gafrc
echo '(component-library ".")' >> gafrc

echo '; zapewniamy unikalnosc rdef jako zalosci poprzez poprzedzanie rdef z nizszego poziomu' ⤸
> gnetlistrc
echo '; atrybutem rdef instancji tego poziomu odzielanej poprzez /' >> gnetlistrc
echo '; czyli dla R1 wlaczonego poprzez schemat podpiety do UKL1 bedzie to UKL1/R1' >> ⤸
gnetlistrc
echo '(hierarchy-uref-separator "/")' >> gnetlistrc
```

```
echo '(hierarchy-uref-mangle "enabled")' >> gnetlistrc

echo "OK"



# sprawdzenie czy plik bazowy istnieje
basename="${1%.sch}"
if [ ! -e "$basename.sch" ]; then
 echo "$basename.sch nie istnieje"
exit
fi



# generowanie PCB
echo "Generate pcb files:"

awk '
 BEGIN {
  FS="[ =]"
 }

 $1=="}" {
  if(is_ok)
   print source, name
  name=""
  source=""
  is_ok=0
 }

 $1=="refdes" {
  name=$2
 }

 $1=="source" && $2!="" {
  source=$2
  is_ok=1
 }
' "$basename.sch" | while read f n; do
 echo -n " * $basename.$n.pcb (based on \"${f%.sch}.pcb\") ... "
if [ -e "${f%.sch}.pcb" ]; then
  sed -e 's#^Element[[]"\([^"]*\)" "\([^"]*\)" "\([^"]*\)"#Element["\1" "\2" "'"$n"'/\3"#g' \
  "${f%.sch}.pcb" > "$basename.$n.pcb" && echo "OK"
else
  echo "ERROR (based file not exist)"
fi
done



# wypisanie informacji końcowej
echo ""
echo "create new pcb file named $basename.pcb and use"
echo " \"File -> Load layout data to paste-buffer\" command"
echo "for insert all generating pcb files into $basename.pcb"
echo "next run \"gsch2pcb $basename.sch\" to prepare ${1%.sch}.new.pcb and $basename.net"
echo "load \"$basename.new.pcb\" and \"$basename.net\" into \"$basename.pcb\""
```

Konfiguracja i dodawanie własnych symboli

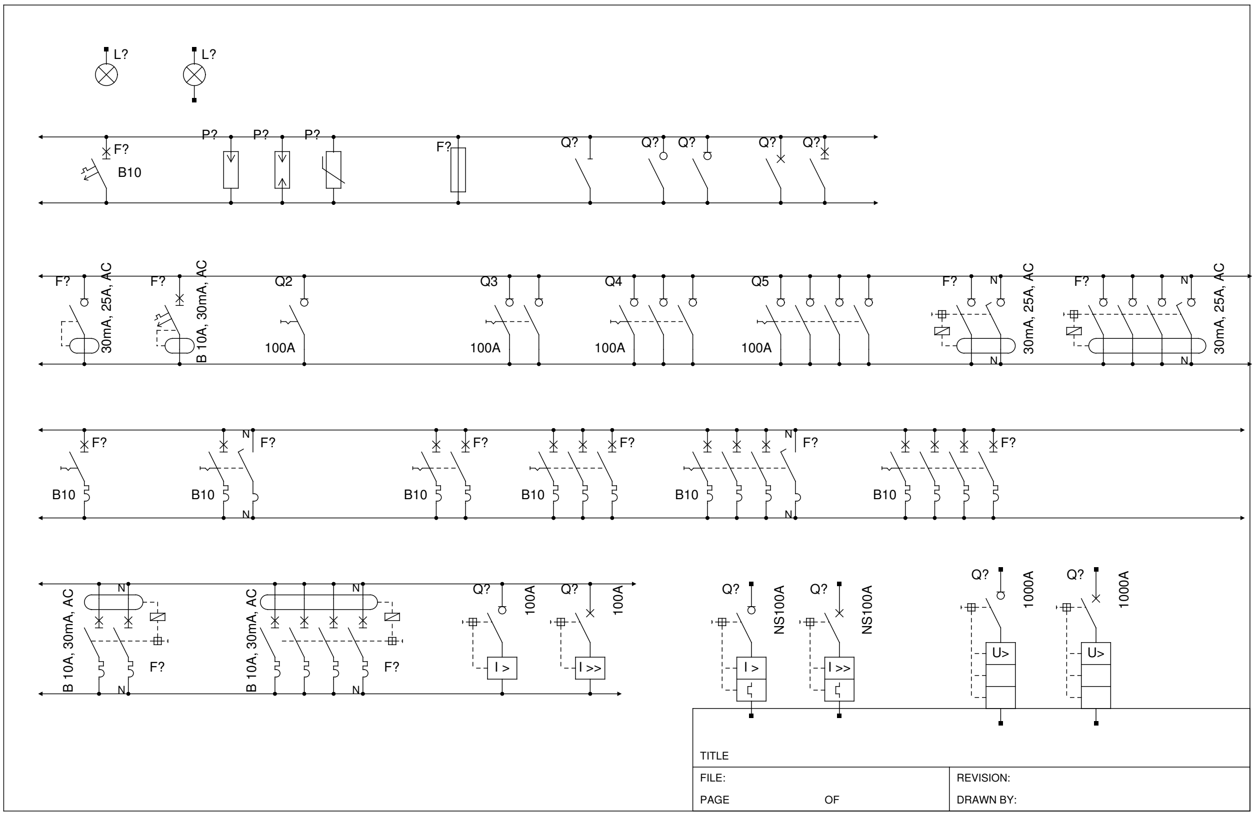
Warto zwrócić uwagę na pliki konfiguracyjne pakietu gEDA - pliki w `~/.gEDA/` stanowią uzupełnienie plików ogólnosystemowych z `/etc/gEDA/`, na szczególną uwagę zasługuje `commonrc` (i `gafrc` - polecam zrobić dowiązanie do `commonrc`) gdzie możemy dodać katalogi do biblioteki symboli (także te wycołane przez autorów - zakomentowane w plikach ogólnosystemowych - polecam w szczególności `(component-library "${GEDADATA}/sym/verilog")`), a także `gschemrc` gdzie możemy m.in. zmienić wariant kolorystyczny programu. Więcej szczegółów znaleźć można w plikach ogólnosystemowych.

Dodatkowe elementy m4 dla pcb (gsch2pcb) możemy umieszczać w pliku `~/.pcb/pcb.inc`. W nowszych wersjach gsch2pcb nie kożysta z pliku `~/.pcb/pcb.inc`, zamiast tego można używa np. `user-config.inc`, warto zatem zsymlinkować te pliki.

Zamieszczam plik pcb.inc[2097] z elementami których mi brakowało (złącza śróbkowe z rastrem 3.5 mm oraz 5mm, przekaźniki z serii RM85, mostek leżący i zmodyfikowane obudowy tranzystorowe TO92, TO220 z możliwością podania nazw pinów, zgodnych z nazwami na symbolu gschem, w kolejności występowania na obudowie; TO220 także w wersji z środkową nóżką wysuniętą w drugą stronę).

Przygotowałem także zbiór symboli[2098] (w formie rozpakowanej[2099]) dla gschem (jfet, p-mosfet, transoptor, optotriak, max485-1, przekaznik, przekaznik dwusekcyjny, przekaznik bistabilny) oraz archiwum[2100] (w formie rozpakowanej[2101]) z dodatkowymi (nie dodanymi do pcb.inc) elementami dla pcb.

W skład wspomnianego zbioru symboli dla gschem wchodzi biblioteka symboli elektrycznych:



Zachęcam także do korzystania z repozytorium symboli na gedasymbols.org[2102].

Nazwy częściej stosowanych footprint'ów

- złącza pinowe {connector.inc}:
  - zwykłe (y * z = xx - ilosc pinów, zazwyczaj y>z) - `CONNECTOR yz`
  - DIP (dwurzędowe, numeracja zgodna z DIP, xx - ilosc pinów) - `CONNECTOR_DIL xx`
  - ZIP (drugi rząd przesunięty o 1/2, xx - ilosc pinów) - `SD xx` {misc.inc}
- obudowy {misc.inc}:
  - SIP (xx - ilosc nóżek) - `CSIL xx` / `SIL xx`
  - DIP (xx - ilosc nóżek, yyy - odstep miedzy rzedami w mils) - `N xxyyy` {dil.inc, aliasem dla N są: NT = J = JD = JG = P, NS jest wariantem o zmniejszonej odległości między nóżkami}
  - okragłe stojące (np. kondensator, yyy - odstep miedzy nóżkami w mils) - `RADIAL_CAN yyyz` (z jest opcjonale, gdy 1 zaznaczona polaryzacja)
  - płaskie stojące (np. kondensator, rezystor, yyy - odstep miedzy nóżkami w mils) - `AXIAL_LAY yyy`

- rezystor:
  - 0.25W - `R025` {misc.inc}
  - regulowany {resistor_adjust.inc}
    - stojacy - `R_AJ_V`
    - lezacy - `R_AJ_H`
- dioda (yyy - odległość między nóżkami) - `DIODE_LAY yyy` {misc.inc}
- dioda LED {misc.inc}:
  - 3 mm - `LED 60`
  - 5 mm - `LED 100`
  - 5 mm z zaznaczona polaryzacja - `T1.75_LED` {newlib/2_pin_thru-hole_packages}
- kwarc (yyy - odległość między nóżkami) - `CRYSTAL yyy` {misc.inc}
- obudowa trojnóżkowa (tranzystorowa / stabilizatorowa / ...) {to.inc}:
  - mala - `TO92`
  - srednia stojaca - `TO220STAND`, z wysuniętą środkową nogą - `TO220STAND_WIDE`
  - srednia lezaca - `TO220LAY`, z wysuniętą środkową nogą - `TO220LAY_WIDE`
  - duża - `TO3_90`, obrócona o 45 stopni `TO3_45`
  - inne takiego typu: `TO5`, `TO126LAY`, `TO126LAY_WIDE`, `TO126STAND`, `TO126STAND_WIDE`
- obudowy `MULTIWATT15`, `MULTIWATT11`, `MULTIWATT8`, `HEPTAWATT`, `PENTAWATT`, `TO220ACSTAND`, `TO251`, `TO264`, `TO247_2`, `TO247`, `TO218` {są to warianty `GENERIC_PL_POWER` z misc.inc zdefiniowane w geda.inc}
- zlacza {connector.inc}:
  - D-Sub (xx - ilosc pinow):
    - meskie - `SUBD_MALE_LAY xx`
    - rzeskie - `SUBD_FEMALE_LAY xx`
  - MTA-156 (pinowe, z jezyczkiem, wieksze, xx - ilosc pinow) - `MTA_156 xx`
  - MTA-100 (pinowe, z jezyczkiem, mniejsze, xx - ilosc pinow) - `MTA_100 xx`
  - USB - `CONN_USB` {gtag.m4}

W nawiasach klamrowych podano nazwy plików w których zdefiniowane są elementy. Wiele z elementów ma krótsze aliasy zdefiniowane w geda.inc. Zamieszczam także zbiór plików[2103] (w formie rozpakowanej[2104]) prezentujących graficznie elementy pcb, wraz z skryptem do ich generowania.

Tekstowe poprawianie plików

Jako że pliki zarówno `gschem`, jak i `pcb` są plikami tekstowymi możemy do ich edycji wykorzystywać także zwykły edytor tekstowy. Jest to przydatne m.in. gdy chcemy aby niektóre z ścieżek zlewały się z wielokątami a inne nie wówczas w pliku pcb dla ścieżek które mają być izolowane od wielokątów ustawiamy ostatni parametr na 0x00000020 a dla tych co maja być połączone na 0x00000000 dwa wcześniejsze parametry odpowiadają za szerokość izolacji (przedostatni i jest to podwojona wartość) oraz szerokość ścieżki (jeszcze wcześniejszy). Uwaga: przed taką edycją warto zrobić kopię zapasową - w szczególności ostatni parametr odpowiada nie tylko za łączenie ścieżek z wielokątami.

Do szybkiej edycji wielu plików, możemy posłużyć się skryptami - np.:
`awk 'BEGIN {FS="="; FIND="device"} $1!=FIND && last!="" {print last} {last=""} $1!=FIND {last=$0}' > OUT.sch < IN.sch` (usunięcie atrybutu device z wszystkich elementów w pliku),
`find -name '*.sch' -o -name '*.pcb' | while read f; do sed 's#TO220STAND#TO220STAND_WIDE#g' < "$f" > /tmp/zam; mv -f /tmp/zam "$f"; done` (zamienia wszystkie elementy TO220STAND na TO220STAND_WIDE).

Podobne skrypty mogą posłużyć sprawdzeniu jakich elementów używaliśmy:
`find -name '*.sch' -exec cat \{\} \; | grep '^C' | cut -f7 -d' ' | sort | uniq` (użyte elementy gschem),
`find -name '*.sch' -exec cat \{\} \; | grep '^footprint=' | cut -f2 -d'=' | cut -f1 -d' ' | sort | uniq` (użyte footprint'y).

Warto tu także wspomnieć o `refdes_renum` nadającym nieponumerowanym elementom (mającym zamiast numeru ?) unikalne numery.

Niekiedy możemy natrafić na problem z nazewnictwem połączeń tranzystorów - zamiast cyfr użyte mogą być oznaczenia literowe C,E,B; aby temu zaradzić zastosować można taki skrypt awk'owy który zamieni oznaczenia:

```
gawk '{
        $0=gensub("T([1-9]+)-E","T\\1-1","g");
        $0=gensub("T([1-9]+)-B","T\\1-2","g");
        $0=gensub("T([1-9]+)-C","T\\1-3","g");
```

```
        print $0;
}' OLD.net > NEW.net
```

Zamieszczam również 3 skrypty służące do przetwarzania plików *.sch i*.pcb:

- gsch2pdf.sh - dokonuje eksportu schematu do pdf'u i png (działa poprawnie gdy schemat nie jest zbyt duży i położony jest w miarę centralnie)
- pcb2pdf.sh - dokonuje eksportu projektu płytki drukowanej do pdf'u
- change_size_in_pcb.sh - modyfikuje rozmiary otworów i pól lutowniczych w projekcie płytki (zapewne będzie wymagał dostosowania do swoich potrzeb)
- build_pcb_version.sh - tworzy wariantowe pliki sch (wybór różnych footprintów dla PCB

GSCH2PDF.SH

```
#!/bin/bash

tmpdir="/tmp"
tmpfile=`mktemp "$tmpdir/gsch2pdf.XXXXXX"`

if [ "$1" = "-size" ]; then
 CONSTANT_SCALE=false;
 shift;
else
 CONSTANT_SCALE=true;
fi

# write script for printing from gschem
init_sch2pdf() {
 if $CONSTANT_SCALE; then
  echo '(output-type "current window")' > ${tmpfile}.scm
  echo '(view-zoom-full)' >> ${tmpfile}.scm
  echo '(paper-size 47.0 111.0)' >> ${tmpfile}.scm
else
  echo '(paper-size 3.5 9.0)' > ${tmpfile}.scm
fi
 cat >> ${tmpfile}.scm << EOF
 (output-orientation "portrait")
 (output-color "disabled")
 (gschem-use-rc-values)
 (gschem-postscript "dummyfilename")
 (gschem-exit)
EOF

cat > ${tmpfile}.rc << EOF
 (component-library ".")
EOF
}

convert_sch2pdf() { (
 set -e
base_name=`basename $1`
base_name="${base_name%.sch}"

cd $2
 # to należy wykonać w katalogu projektu ze względu na możliwość występowania tam ↩
 plików pomocniczych
 gschem -q -o ${tmpfile}.ps -s ${tmpfile}.scm -r ${tmpfile}.rc "$base_name.sch" &> /dev/null

cd $tmpdir
 #grep -v 'scale$' ${tmpfile}.ps | grep -v 'translate$' |
 # sed 's#2 setlinecap#2 setlinecap\n0.04 0.04 scale\n-27500 -33000 translate#g' > ${tmpfile}
```

```
 # mv ${tmpfile} ${tmpfile}.ps

if $CONSTANT_SCALE; then
  ps2eps -O -n -P -l -q ${tmpfile}.ps
else
  ps2epsi ${tmpfile}.ps ${tmpfile}.eps
fi
 epstopdf ${tmpfile}.eps* -o ${tmpfile}.pdf

# optional write png
 if [ "$3" = "true" ]; then
  # convert ${tmpfile}.pdf "$2/$base_name.png"
  pdftoppm ${tmpfile}.pdf ${tmpfile}
  pnmtopng ${tmpfile}-*1.ppm > "$2/$base_name.png" 2> /dev/null
fi

# move pdf to workdir
 mv ${tmpfile}.pdf "$2/$base_name.pdf"
) }

base_pwd=$PWD
for f in "$@"; do
 init_sch2pdf
convert_sch2pdf "$f" "$base_pwd" "true" || echo "sch2pdf fail for $f" > /dev/stderr
rm ${tmpfile}*
done
```

PCB2PDF.SH

```
#!/bin/bash

tmpdir="/tmp"
tmpfile=`mktemp "$tmpdir/pcb2pdf.XXXXXX"`

init_page_list() {
 # pobranie spisu treści dla funkcji get_page_no
 CONTENT=`pstops -q '1000:0' $1 | ps2ascii | sed -e 's#[.] *#.#g' -e 's#[^a-zA-Z0-9.-]#\n#g'`
}

get_page_no_old() {
 page=`echo "$CONTENT" | egrep '^[0-9]+[.]'"$1"'$' | cut -f1 -d'.'`
echo $(( $page - 1 ))
}

get_page_no() {
 page=`grep "%%Page: $1\b" ${tmpfile}-1.ps | cut -f3 -d' '`
if [ "$page" != '' ]; then
  echo $(( $page - 1 ))
  return 0
 else
  echo 0
  return 1
 fi
}

convert_pcb2pdf() { (
 set -e
cd $tmpdir
```

```
base_name=`basename $1`
base_name="${base_name%.pcb}"

# drukowanie PCB jako mirror dla potrzeb termotransferu (str. ”solder”):
 pcb -x ps --media A4 --mirror --psfile ${tmpfile}-1.ps "$2/$base_name.pcb" >/dev/null

# pobranie numeru strony
 info="bottom"
if ! pages=`get_page_no "$info 2"`; then
  info="top"
  if ! pages=`get_page_no $info`; then
   info="bottom"
   pages=`get_page_no $info`
  fi
 fi
 info="$info ($pages)"
pstops -q "1000:$pages" ${tmpfile}-1.ps ${tmpfile}-2a.ps

# drukowanie PCB dla potrzeb iformacji o rozmieszczeniu elementów
 pcb -x ps --media A4 --psfile ${tmpfile}-1.ps "$2/$base_name.pcb" >/dev/null
init_page_list ${tmpfile}-1.ps
pages=`get_page_no "topsilk"`
pages="$pages,`get_page_no "bottomsilk"`"
pages="$pages,`get_page_no "fab"`"
pages="$pages,`get_page_no "topassembly"`"
pages="$pages,`get_page_no "bottomassembly"`"
pstops -q "1000:$pages" ${tmpfile}-1.ps ${tmpfile}-2b.ps

gs -q -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile="$2/$base_name.pcb.pdf" \
-f ${tmpfile}-2a.ps ${tmpfile}-2b.ps

echo "$base_name: $info + addionals ($pages)"
) }

base_pwd=$PWD
for f in "$@"; do
 convert_pcb2pdf "$f" "$base_pwd" "true" || echo "dia2pdf fail for $f" > /dev/stderr
rm ${tmpfile}*
done
```

CHANGE_SIZE_IN_PCB.SH

```
#!/bin/bash

change_size() {
 external_from=$1
 internal_from=$2
 external_to=$3
 internal_to=$4
 sed -e "s#Pin\[\([0-9-]*\) \([0-9-]*\) $external_from \([0-9-]*\) \([0-9-]*\) ↩
 $internal_from#Pin[\1 \2 $external_to \3 \4 $internal_to#"
}

for f in "$@"; do
 if [ -f $f ]; then
  if [ -f "$f.bck" ]; then
   echo "WARNING: Old backup file exist - will be used existing $f.bck as input"
  else
   mv "$f" "$f.bck"
```

```
  fi
  cat "$f.bck" |
    change_size 5500 3000 6000 2800 | # otworki 0.055/0.030 (AXIAL_LAY) inch na 0.060/0.028 ⤸
    inch (0.7 mm)
    change_size 5500 2500 6000 2800 | # otworki 0.055/0.030 (AXIAL_LAY cust.) inch na 0.060/0.⤸
    028 inch (0.7 mm)
    change_size 6000 3000 6000 2800 | # otworki 0.060/0.030 (DIODE_LAY) inch na 0.060/0.028 ⤸
    inch (0.7 mm)
    change_size 6800 3800 6400 3200 | # otworki 0.068/0.038 (R025) inch na 068/0.032 inch (0.8 ⤸
    mm)
    change_size 6800 3300 6400 3200 | # otworki 0.068/0.038 (R025 cust.) inch na 068/0.032 ⤸
    inch (0.8 mm)
    change_size 6000 3800 6500 3600 | # otworki 0.060/0.038 (CONNECTOR) inch na 0.065/0.036 ⤸
    inch (0.9 mm)
    change_size 6000 3300 6500 3600 | # otworki 0.060/0.033 (CONNECTOR cust.) inch na 0.065/0.⤸
    036 inch (0.9 mm)
    change_size 7000 3500 8000 4700 | # otworki 0.070/0.035 (CONNECTOR_AWG35) inch na 0.080/0.⤸
    047 inch (1.2 mm)
    change_size 9000 5000 9500 6000 > "$f" # otworki 0.090/0.050 (CONNECTOR_AWG50) inch na 0.⤸
    095/0.060 inch (1.2 mm)
 fi
 #    change_size 6000 3500 6000 3400
 #    change_size 5000 2000 6800 3300

#    # modyfikujemy informacje nagłówkowe
#    sed -i -e 's#DRC\[.*$#DRC[1000 1000 1400 1000 2000 1400]#g' \
#      -e 's#Flags(.*$#Flags("showdrc,rubberband,nameonpcb,autodrc")#g' \
#      -e 's#Groups(.*$#Groups("1,2,3,s:4,c:5:6:7:8:s:c")#g' \
#      -e 's#Styles\[.*$#Styles["Signal,1500,6000,2800,1000:Power,2500,7000,4000,1000:Fat,4000,⤸
13000,8500,1000:Skinny,800,3600,2000,1000"]#g'
#      "$f"

# ROZMIARY MINIMALNE:
# - ścieżki 14/15 mil (0.35/0.38 mm)
# - pole lutownicze wokół otworu (Pin) (65-36)/2 = 14.5 mil (0.36 mm)
# - odstępy między ścieżkami 10 mil (0.254mm)
# - otwór 20/28 mil (0.5/0.7 mm)
done;
```

BUILD_PCB_VERSION.SH

```
#!/bin/bash

# skrypt służy do budowania wersji jednej z kilku możliwych wersji PCB na podstawie ⤸
odpowiednio oprzygotowanego pliku sch
#    * dla każdej z wersji podaje się footprint_{numerwersji}
#    * dla elementów mających taki sam footprint w WSZYSTKICH wersjach wystarczy podać tylko ⤸
atrybut footprint

nazwa_projektu=$1
numer_wersji=$2

# zamieniamy footprint_{numerwersji} na footprint
sed 's#^footprint_'"$numer_wersji"'=#footprint=#g' < $nazwa_projektu_din.sch > ⤸
"$nazwa_projektu_$numer_wersji.tmp.sch"

# wywalamy pozostale footprint_
awk 'BEGIN {FS="="} $numer_wersji ~ /^footprint_/ && last!="" {print last} {last=""} ⤸
$numer_wersji !~ /^footprint_/ {last=$0}' \
```

```
< "$nazwa_projektu_$numer_wersji.tmp.sch" > "$nazwa_projektu_$numer_wersji.sch"

rm -f "$nazwa_projektu_$numer_wersji.tmp.sch"
gsch2pcb "$nazwa_projektu_$numer_wersji.sch"
```

Linki

Zobacz w Sieci: gEDA[2105], PCB[2106], Using gEDA (w LinuxFocus)[2107]

QCad

Jest to całkiem przyjemny program typu CAD, służący do tworzenia rysunków technicznych takich jak projekty architektoniczne (np. rzuty pięter), projekty elementów mechanicznych, schematy instalacji elektrycznych, czy nawet schematy elektroniczne lub blokowe (aczkolwiek tu lepiej u żyć bardziej specjalistycznego oprogramowania).

Przy drukowaniu projektów warto szczególną uwagę zwrócić na rozmiar papieru zdefiniowanych w właściwościach bieżącego rysunku (menu edycja) - to ona odpowiada za wielkość kartki którą przykładamy na podglądzie wydruku.

Przedstawiam także zestaw symboli, mojego autorstwa, przydatnych przy projektowaniu instalacji i rozdzielnic elektrycznych - qcad-elektryczne.tgz[2108] (w formie rozpakowanej[2109])

Dia i różne schematy

Jest to całkiem program do tworzenia diagramów i schematów. Umożliwia tworzenie m.in:

- diagramów blokowych
- diagramów logicznych (z użyciem symboli ALU, MUX, bramek, ...)
- diagramów przepływu dla algorytmów (Flowchart[2110])
- grafów stanu dla automatów (State diagram[2111])
- schematów UML[2112]
- schematów telekomunikacyjnych
- schematów elektronicznych (jednak tutaj warto użyć dedykowanego oprogramowania takiego jak `kicad`, czy `gschem` - posiada ono na ogół znacznie więcej symboli oraz umożliwia tworzenie netlist i przejście do projektowania pcb)
- schematów elektrycznych (tutaj także można użyć oprogramowania takiego jak `kicad`, czy `qcad`)

Ponadto dzięki bliskości formatu dia do svg narzędzie to ładnie współpracuje z `inkscape` (możemy go użyć np. do przygotowywania elementów używanych w dia), a także umożliwia eksport do tego formatu. Główną niedogodnością w dia jest brak możliwości swobodnego obracania elementów.

Możemy instalować także w `~/.dia/` dodatkowe "shape" do tego programu, trochę gotowych znaleźć można na Dia Shape Repository[2113], Shapes for Dia[2114] (te i nie tylko w formie paczki deb[2115]). Poprzez edycję pliku konfiguracyjnego `~/.dia/sheets/Assorted.sheet` możemy ustalać zawartość grupy "Mieszane" (wyróżnia się tym iż zawsze jest na górze menu) - przykład[2116].

Zamieszczm także skrypt do automatycznej konwersji schematów dia na pliki w formatach pdf i png - dia2pdf.sh

dia2pdf.sh

```
#!/bin/bash

tmpdir="/tmp"
tmpfile=`mktemp "$tmpdir/pcb2pdf.XXXXXX"`

convert_dia2pdf() { (
 set -e
cd $tmpdir

base_name=`basename $1`
```

```
base_name="${base_name%.dia}"

dia -e ${tmpfile}.eps "$2/$base_name.dia" &> /dev/null

epstopdf ${tmpfile}.eps -o ${tmpfile}.pdf

# optional write png
 if [ "$3" = "true" ]; then
  # convert ${tmpfile}.pdf "$2/$base_name.png"
  pdftoppm -r 60 ${tmpfile}.pdf ${tmpfile}
  pnmtopng ${tmpfile}-*1.ppm > "$2/$base_name.png" 2> /dev/null
fi

# move pdf to workdir
 mv ${tmpfile}.pdf "$2/$base_name.pdf"
) }

base_pwd=$PWD
for f in "$@"; do
 convert_dia2pdf "$f" "$base_pwd" "true" || echo "dia2pdf fail for $f" > /dev/stderr
done

rm ${tmpfile}*
```

GnuPlot

— rysowanie danych z plików (wynik.dat - kolumna 1 jako x kolumna 9 jako y) i zadanej funkcji (sinus) na jednym wykresie - `plot 'wynik.dat' using 1:9, sin(x)`
— tytuł wykresu w legendzie - `plot sin(x) title "sinus"`
— nieumieszczanie wykresu w legendzie - `plot sin(x) notitle`
— wyłączanie wyświetlania legendy - `unset key`
— legenda po prawej na dole - `set key right bottom`

— zakres osi od -10 do 25 - `set yrange [-10:25]`, `set xrange [-10:25]`
— automatyczny zakres osi - `set yrange [ * : * ]; set xrange [ * : * ];`
— opis osi - `set xlabel "na osi x mamy ..."; set ylabel "na osi y mamy ...";`
— skala logarytmiczna na osi x - `set log x`
— podziałki na osi X od 0 co 0.5 do 16 - `set xtics 0,0.5,16`
— ukrycie podziałek - `unset tics`
— pokazanie siatki - `set grid`

— tytuł wykresu - `set title "tytuł wykresu"`
— ukrycie ramki otaczającej wykres - `unset border`

— definiowanie własnych funkcji oraz instrukcja warunkowa - `sp(x,a)=x<1 && x>-1 ? x/a : x<=-1 ? -1.0/a : 1.0/a;` oraz użycie `plot [-6:6] [-1.7:1.7] sp(x,1)` (dodatkowo ustawianie zakresów osi wewnątrz komendy plot)
— wiele komend plot na jednym wykresie - `set multiplot` (zakończenie: `unset multiplot`)
— rysowanie w trybie parametrycznym - `set parametric; set trange [-1.7:1.7]; plot 1.9,t with dots linecolor 1 notitle` (dodatkowo ustawiamy styl linii na kropkowanie)
— kółko w trybie parametrycznym - `set parametric; plot [0:2*pi] sin(t),cos(t); unset parametric` (także zakończenie parametrycznego)

— zapis do PostScriptu - `set terminal postscript color; set output "wykresy.ps"`
— zapis do SVG - `set term svg; set output "wykresy.svg"` po narysowaniu warto zrobić `! echo "</svg>" >> "wykresy.svg"` co poprawnie zakończy plik (wykonanie zewnętrznej komendy `echo ...`)
— zapis do "epslatex" - `set terminal epslatex color; set output "plot.tex"; plot f(x) title $V_{sum}-2\\varphi$; set output; set terminal pop;`, potem `epstopdf plot.ps; pdflatex plot-make.tex`, gdzie "plot-make.tex":

```
\documentclass[10pt]{article}
\usepackage{graphicx}
\usepackage[paperwidth=330px,paperheight=260px,margin=0px,lmargin=-50px]{geometry}
\begin{document}\thispagestyle{empty}\input{plot.tex}\end{document}
```

potem ewentualnie także `pdftoppm plot-make.pdf plot-make; pnmtopng plot-make-000001.ppm > plot-make.png`; podobny efekt można uzyskać generując zwykły eps i korzystając z psfrag

Zamieszam też przykładowy skrypt gnuplot-owy obrazujący zagadnienia mocy czynnej, biernej i tangensa fi.

Zobacz w Sieci: Quick Reference Card for gnuplot[2117] - kilkustronicowa broszurka z najważniejszymi poleceniami, Wstęp do gnuplota[2118], Gnuplot Manual[2119] - oficjalny podręcznik [en], LaTeX and the Gnuplot Plotting Program[2120]

MOCE.PLOT

```
#!/usr/bin/gnuplot -p

# napiecie (napięcie skuteczne = 1V, amplituda = sqrt(2)V)
Usk     = 1
u(t)    = sin(t) * Usk * sqrt(2)

# prad (prąd skuteczny = 1A,    amplituda = sqrt(2)A)
Isk     = 1
i(t,fi) = sin(t+fi) * Isk * sqrt(2)


# moc chwilowa [VA]
p(t,fi) = i(t,fi)*u(t)


# moc pozorna (1VA)
S       = Isk * Usk

# moc czynna = calka po okresie (T=2*pi) z p(t,fi)
P(fi)   = S * cos(fi)

# moc bierna = sqrt (S^2 - P^2)
Q(fi)   = S * sin (fi)


set term wxt size 900,680
set multiplot layout 3,2 rowsfirst

set key left bottom at graph 0.13,0.06 # samplen 2 spacing .7 font ",8"
set border 3; set xtics nomirror; set ytics nomirror
set xzeroaxis

set title "obc. indukcyjne, cos(fi) = 0.9, tg(fi) = 0.48, fi = 0.45"; fi=0.4510
plot [0 : 2*pi] u(x) title "u(t)", i(x,fi) title "i(t)", p(x,fi) title "p(t)"

set title "obc. pojemnościowe, cos(fi) = 0.8 tg(fi) = -0.75, fi = -0.64"; fi=-0.6435
plot [0 : 2*pi] u(x) title "u(t)", i(x,fi) title "i(t)", p(x,fi) title "p(t)"

set title "obc. indukcyjne, cos(fi) = 0.6, tg(fi) = 1.3, fi = 0.93"; fi=0.9273
plot [0 : 2*pi] u(x) title "u(t)", i(x,fi) title "i(t)", p(x,fi) title "p(t)"

set title "moce w funkcji cos(fi), zakres indukcyjny";
set key right center at graph 0.97,0.55
plot [1 : 0] S title "pozorna" , P(acos(x)) title "czynna", Q(acos(x)) title "bierna"
unset xrange # trzeba posprzatac po odwroconym zakresie osi x ...
```

```
set size 1,0.33; set origin 0.0,0.0;
set title "moce w funkcji przesunięcia fazowego fi";
set key right bottom at graph 0.95,0.05
plot [-pi/2 : pi/2] S title "pozorna" , P(x) title "czynna", Q(x) title "bierna"

unset multiplot
```

Biblioteczki dla uC

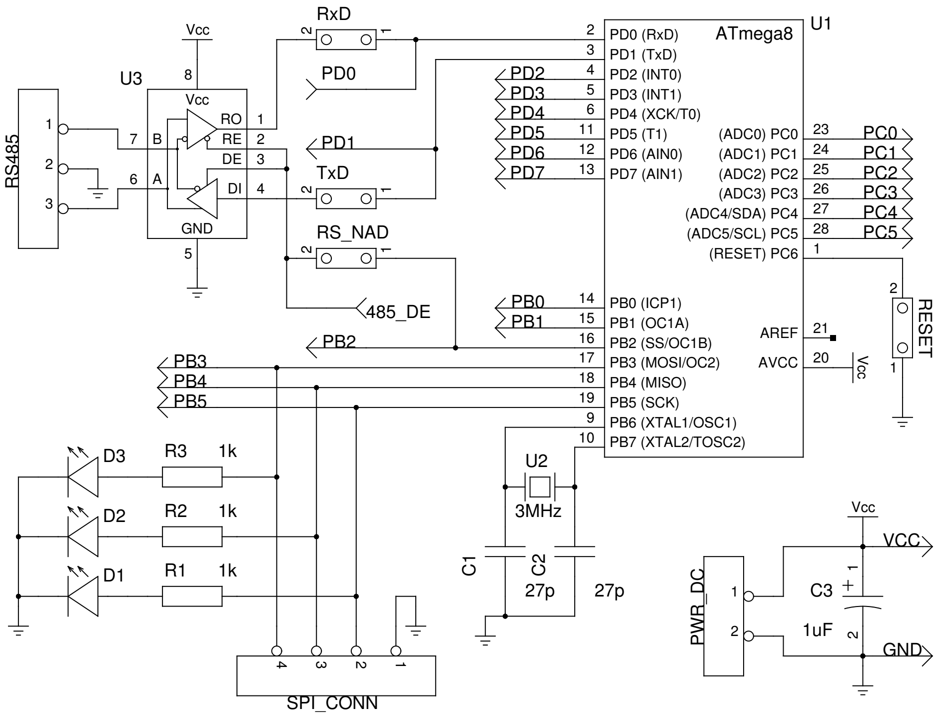
Dział ten zawiera zbiór użytecznych, powtarzalnych fragmentów kodu w C dla projektów opartych na mikrokontrolerach AVR:

- lcd.c
- onewire.c
- opoznienia.c

A także projekt standardowego modułu elektroniki opartej o mikrokontroler Atmega8 - zawierający złącze zasilania, interfejs RS485, złącze SPI (programowanie ISP), kwarc, diody diagnostyczne:

- schemat ideowy[2121](PDF: `http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/atmega8_uart_spi.pdf`)
- projekt płytki[2122](PDF: `http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/atmega8_uart_spi.pcb.pdf`)
- symbol[2123]
- dodatkowo także projekt płytki[2124](PDF: `http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/atmega8_dev_board.pcb.pdf`) w wersji "deweloperskiej" - z wyprowadzonymi wszystkim sygnałami z Atmega8 na goldpiny

`atmega8_uart_spi.sch:`



Modbus dla AVR (oparty na FreeMODBUS[2125] - open-source implementacji Modbus ASCII/RTU i TCP):

- Makefile[2126] - plik Makefile do budowy projektów wykorzystujących modbus
- modbus.tar[2127] - bazowe pliki FreeMODBUS 1.5.0 (2010-05-06) rozpowszechniane na licencji typu BSD, z zmodyfikowanym `include/mbconfig.h`
- modbus.c[2128] i port.h[2129] - pliki związane z portem FreeMODBUS na Atmega8

ONEWIRE.C

```
/**
 * Plik stanowi zestaw funkcji do obsługi interfejsu 1-wire
 * do wykorzystania zarówno w mikrokontrolerach jak i sprzęcie PC
 * konieczne jest wstawienie odpowiedniej wersji definiowania funkcji:
 *      onewire_send_zero() onewire_send_jeden() onewire_read() - patrz poniżej
 *
 * a także włączenie "util/delay.h"
 * lub odpowiednika (np. dla PC - patrz onewire_get_rom.c)
**/

/* PRZYKŁADY DEFINICJI FUNKCJI ZALEŻNYCH OD IMPLEMENTACJI:
1) dla portu LPT w komputerze PC:
 #define onewire_send_zero() outb(0x00, 0x37a)
 #define onewire_send_jeden() outb(0x04, 0x37a)
 #define onewire_read() (inb(0x37a) & 0x04) >> 2

2) dla mikrokontrolera z pojedyńczym interfejsem na PB6
 #define onewire_send_zero() DDRB |= 0x40; PORTB &= 0xbf
 #define onewire_send_jeden() PORTB |= 0x40; DDRB &= 0xbf
 #define onewire_read() (PINB & 0x40) >> 6

3) dla mikrokontrolera z 8-cioma interfejsami na PORTB
 #define MULTI_BIT 6
 #define onewire_send(byte) DDRB = ~byte; PORTB = byte
 #define onewire_read()          PINB
*/

#ifdef MULTI_BIT
# define onewire_send_zero()  onewire_send(0x00)
# define onewire_send_jeden() onewire_send(0xff)
#endif

/// funkcja resetująca szyne onewire
void onewire_reset() {
 cli();

onewire_send_zero();
 _delay_ms( 0.5 ); // 500 us
 onewire_send_jeden();
 sti();
 _delay_ms( 10 );
}

/// funkcja przesyłająca pojedyńczy bit poprzez onewire
///    dla trybu jednoliniowego wysyła bit 1 gdy b!=0
///    dla trybu wieloliniowego wysyła na kolejnych liniach kolejne bity b
uint8_t onewire_bit(uint8_t b) {
 cli();

onewire_send_zero();
#ifdef MULTI_BIT
 onewire_onewire_send(b);
#else
 if (b) onewire_send_jeden();
#endif
 _delay_us( 20 );
 b = onewire_read();
 _delay_us( 50 );
```

```
 onewire_send_jeden();

sti();
 return b;
}

/// funkcja przesyłająca pojedyńczy bajt poprzez onewire
///    dla trybu jednoliniowegozwraca otrzymany bajt odpowiedzi
uint8_t onewire_byte(uint8_t b_in) {
#ifdef MULTI_BIT
 uint8_t i, j, tmp;
 for (j=0; j<8; j++) {
  // powielenie bierzącego bitu na cały bajt
  if (b_in & (1<<j))
   tmp=0xff;
  else
   tmp=0x00;

  // obsługa interfeju
  onewire_bit(tmp);
 }
#else
 uint8_t c;
 for (c=8; c; c--)
  b_in = ( onewire_bit( b_in & 0x01 ) << 7 ) | ( b_in >> 1 );
  // w najstarszy bit zapisuje zczytana wartosc
  // wykonuje binarne lub z
  // poprzednia wartoscia b przesunieta o 1 w prawo
#endif
 return b_in;
}

/// funkcja przesyła bajt w trybie liniowym,
/// ale wartości dla poszczególnych bitów brane są z osobnych komórek
/// tabeli a nie poprzez powielenie bitu pojedyńczego bajtu
/// jak ma to miejsce w onewire_byte()
#ifdef MULTI_BIT
void onewire_byte_array(const uint8_t *b_in) {
 uint8_t i, j, tmp1, tmp2;
 for (j=0; j<8; j++) {
  // konstrukcja bajtu (tmp1) zlozonego z aktualnego bitu (j <=> tmp2) każdej linii
  tmp1=0x00;
  tmp2=(1<<j);
  for (i=0; i<MULTI_BIT; i++) {
   tmp1 |= (b_in[i] & tmp2) << i;
  }

  // obsługa interfeju
  tmp1 = onewire_bit(tmp1);
 }
}
#endif

/// funkcja wysyłając same jedynki odbiera odpowiedz z onewire_get_rom
///    dla trybu jednoliniowego jest to alias do onewire_byte(0xff)
///        i wynik zwracany jest przez return
///    dla trybu wieloliniowego wynik zwracany jest w podanej jako argument
///        tablicy o wielkości określonej przez MULTI_BIT
#ifdef MULTI_BIT
void onewire_reply_array(uint8_t *b_out) {
```

```
 uint8_t i, j, tmp;
 for (j=0; j<8; j++) {
  // obsługa interfeju
  tmp = onewire_bit(0xff);

  // analiza otrzymanej odpowiedzi
  // (podział bajtu na bity do osobnych komórek tabeli)
  for (i=0; i<MULTI_BIT; i++) {
   b_out[i] |= (tmp & (1<<i)) << j;
  }
 }
}
#else
# define onewire_reply() onewire_byte(0xff)
#endif

/// funkcja przesyła komendę onewire
void onewire_command(uint8_t command, const uint8_t *adres) {
 onewire_reset();

if (adres == NULL) {
  onewire_byte( 0xcc ); // skip room
 } else {
  char c;
  onewire_byte( 0x55 ); // match rom
  for (c=8; c; c--) {
   onewire_byte(*adres);
   adres++;
  }
 }
 onewire_byte(command);
}

#ifdef MULTI_BIT
/// funkcja przesyła komendę poprzez wieloliniowy onewire, gdzie
/// na poszczególnych liniach wystawiane są niezależne adresy urządzeń
/// adresy podawane są jako tablica 8*MULTI_BIT w postaci:
/// adr0[7], ..., adr0[0], adr1[7], ..., adr1[0], ...
void onewire_command_array(uint8_t command, const uint8_t *adres) {
 uint8_t i, j, addr_tmp[8];
 onewire_reset();

onewire_byte( 0x55 ); // match rom
 for (i=0; i<8; i++) {
  for (j=0; j<MULTI_BIT; j++) {
   addr_tmp[j]= adres[j*8+i];
  }
  onewire_byte_array(addr_tmp);
 }

onewire_byte(command);
}
#endif
```

LCD.C

```
/**
 * Plik stanowi zestaw funkcji do obsługi wyświetlacza na bazie HD44780
```

```
 * do wykorzystania zarówno w mikrokontrolerach jak i sprzęcie PC
 * konieczne jest wstawienie odpowiedniej wersji definiowania funkcji:
 *      lcd_send() - patrz poniżej
**/

/*
#ifndef lcd_send
 //#define lcd_send(byte) outb(byte, 0x378)
 #define LCD_E              0x10
 #define LCD_RS_DATA     0x20
 #define LCD_RS_CMD       0x00
 #define LCD_DATA_OFFSET 0
 #define lcd_send(byte) PORTC = byte
#endif
*/

/// wysłanie komendy lub danych z obsługą przełączania E, RS
/// gdy mode == 0x00 to komenda
/// gdy mode == 0x10 to dana
void lcd_switch_and_send(uint8_t val, uint8_t mode) {
 val &= 0x0f;

lcd_send(mode); // RS = mode E = 0
 lcd_send((val << LCD_DATA_OFFSET) | LCD_E | mode); // E =1
 _delay_ms(1);
 lcd_send(mode); // E = 0
}

/// wysłanie komendy lub danych z obsługą przełączania E, RS
/// gdy jmamy 4 bitową szynę
void lcd_switch_and_send_4bit(uint8_t val, uint8_t mode) {
 lcd_switch_and_send((val >> 4) & 0x0f,    mode);
 lcd_switch_and_send(val & 0x0f, mode);
}

/// pomcnicze makra związane z wysyłaniem komend / danych
#define lcd_send_comand(val) lcd_switch_and_send_4bit(val, LCD_RS_CMD)
#define lcd_send_data(val)   lcd_switch_and_send_4bit(val, LCD_RS_DATA)

/// pomocnicza funkcja obsługująca wysyłanie napisu
void lcd_send_string(const char *napis, uint8_t dlugosc) {
 uint8_t i;
 for (i=0; i<dlugosc; i++) {
  lcd_send_data( napis[i] );
 }
}

/// pomocnicze makra związane z ważnymi komendami
// Function Set (2 linie 4 bit) = 0010, 0010 10**
#define lcd_set_4bit_2line() lcd_switch_and_send(0x20, LCD_RS_CMD); lcd_send_comand(0x28)
// On-Off Control (wyświetlacz włączony, kursor wyłączony) = 0000 1100
#define lcd_set_display_on_cursor_off() lcd_send_comand (0x0c);
// Entry Mode Set (przesówanie w prawo przy zapisie do DD RAM) = 0000 0110
#define lcd_set_move_right() lcd_send_comand (0x06);

#define lcd_init_4bit_2line() \
 lcd_set_4bit_2line();\
 lcd_set_display_on_cursor_off();\
 lcd_set_move_right();
```

```
// Clear Display = 0000 0001
#define lcd_clear() lcd_send_comand(0x01)
// First line (Return Home) = 0000 0010
#define lcd_first_line() lcd_send_comand(0x02)
// Second line (Return Home 2) = 1100 0000
#define lcd_second_line() lcd_send_comand(0xc0)
```

OPOZNIENIA.C

```
/**
 * plik biblioteczki opóźniającej
 * zawiera funkcję _delay_cs(uint8_t) wstawiającą opóźnienie do 2.56 sekundy
 * inkluduje także util/delay.h z funkcjami
 *    void _delay_ms (double ms)
 *      -> MAX: 262.14 ms / F_CPU in MHz
 *           for 22MHz MAX=11.915
 *    void _delay_us (double us)
 *      -> MAX: 768 us / F_CPU in MHz
 *           for 22MHz MAX=34.909
 **/

#ifndef F_CPU
 #error "F_CPU must be declared before use opoznienia.c"
#endif

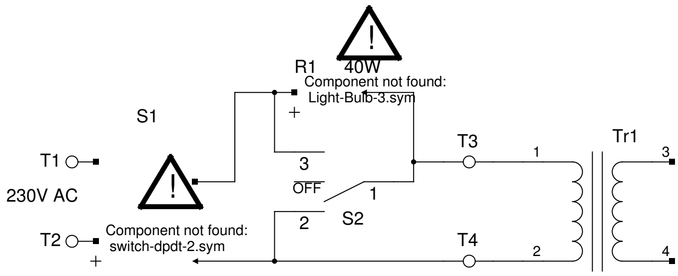
#if F_CPU > 22000000L
 #error "F_CPU must be < 22000000L (22MHz)"
#endif

#include <inttypes.h>
#include <util/delay.h>

/** pętla opóźniająca w centy-sekundach (0.01s) - bez gwarancji precyzji
 maksymalna wartość opóźnienia to 256 cs czyli 2.56 sekund **/
void _delay_cs(uint8_t licz) {
 for (; licz>0; licz--){
  _delay_ms(10);
 }
}
```

Układ ograniczający prąd rozruchowy

Niektóre urządzenia charakteryzują się poborem znacznego prądu w momencie załączenia napięcia, dotyczy to w szczególności transformatorów dużej mocy. W obwodach zasilających takie urządzenia należy stosować odpowiednie zabezpieczenia zwłoczne. Jednak gdy instalacja posiada zbyt szybkie zabezpieczenia przydatny może być układ ograniczający prąd rozruchowy. Działanie takiego układu polega na rozruch poprzez szeregowo włączony z urządzeniem rezystor który jest następnie bocznikowany. Koncepcję takiego układu wykorzystującego jako rezystor żarówkę (usuwa to problem płonących rezystorów gdy mają za małą moc) przedstawiam poniżej.



S1 - main switch
S2 - mode switch:
1) 1 - 2: Tr1 is OFF, lightbulb R1 is full ON
2) 1 - OFF: Tr1 is powerd via R1 (limiting current)
3) 1 - 3: Tr1 is full ON

S1 - wyłącznik główny
S2 - przełącznik trybu:
1) 1 - 2: Tr1 WYŁĄCZONY, żarówka R1 w pełni WŁĄCZONA
2) 1 - OFF: Tr1 jest zasilany poprzez R1 (ograniczenie prądu)
3) 1 - 3: Tr1 w pełni WŁĄCZONY

Zobacz także zdjęcia wykonanego urządzenia[2130]

Generator sygnałów

Opcje linii poleceń

Program obsługuje typowe opcje w stylu unixowym:

- `-o` tryb wyjściowy, możliwe wartości:
  - dsp - wyjście realizowane z użyciem karty dźwiękowej poprzez plik wskazany /dev/dsp
  - alsa - wyjście realizowane z użyciem karty dźwiękowej poprzez wskazane urządzenie systemu ALSA
  - uC - wyjście realizowane z użyciem modułu elektronicznego (mikrokontrolera) poprzez port szeregowy
- `-O` urządzenie wyjściowe (np. "default" dla ALSA, "/dev/dsp" dla dsp, "/dev/ttyS0" dla uC)
- `-r` częstotliwość próbkowania na urządzeniu wyjściowym w Hz (uwaga wartości nie typowe dla kart audio mogą być re-samplowane programowo przez sterownik)
- `-d` plik (lub "-" dla standardowego wyjścia) w którym ma być zapisany pojedynczy okres wygenerowanego sygnału, może służyć do zobrazowania sygnału z użyciem gnuplota poprzez `echo 'plot "/wskazany/plik"' | gnuplot -persist`
- `-h` wyświetlenie treści pomocy

Wszystkie opcje poza `-d` i `-h` są obowiązkowe. Po opcjach należy wskazać używany plik konfiguracyjny opisujący sygnał. Przykład wywołania:

```
./generator -d /tmp/log -o dsp -r 48000 -O /dev/dsp0 config.txt
```

Definiowanie sygnału - plik konfiguracyjny

Tryb wyjścia oraz częstotliwość próbkowania podawane są jako parametry linii poleceń, natomiast sygnał opisywany jest w pliku tekstowym o następującym formacie:

- pierwsza linia zaczyna się od słowa kluczowego `TIME` i zawiera dwa parametry (rozdzielane pojedynczą spacjami):
  - liczbę okresów przez którą będzie powtarzany sygnał (0 oznacza bez ograniczeń)
  - czas trwania pojedynczego okresu (wyrażony w sekundach, mogą być także zastosowane oznaczenia typu 10m i 10u co odpowiada 10 ms oraz 10 us)
- druga linia zaczynająca się od słowa kluczowego określającego tryb działania i zawierająca stosowne do niego parametry może mieć postać:
  - `SQUARE time` - sygnał kwadratowy, gdzie stan wysoki trawa przez podany czas (parametr `time`)
  - `SIN period` - sygnał sinusoidalny, o podanym okresie (parametr `period`)
  - `SAW_R time` - sygnał piłokształtny, opadający przez podany czas (parametr `time`)

  - `SAW_L time` - sygnał piłokształtny, narastający przez podany czas (parametr `time`)
  - `TRIANGLE timeA timeB` - sygnał trójkątny, narastający i opadający przez podany czas (odpowiednio parametry `timeA` i `timeB`)
  - `FUNCTION` - dowolnie definiowana funkcja od argumentu x który w trakcie okresu zmienia się od 0 do 2pi
  - `PCM` - dowolnie definiowany przebieg PCM (każda kolejna linia to pojedyncza próbka sygnału, podana jako liczba od -1 do 1)

  atrybuty określające czas wyrażone są w sekundach, podobnie jak w linii TIME mogą posiadać miejsca po przecinku oraz określenia u i m oznaczające odpowiednie mnożniki SI.
- atrybut określający okres trwania sygnału z linii TIME ma większą siłę od określeń z linii definiującej sygnał (nadmiar sygnału zostanie ucięty, a niedomiar dopełniony zerami)
- w makrach posiadających jeden parametr może zostać on pominięty (zostanie wtedy użyty atrybut określający długość sygnału z linii TIME)

Źródła

Program napisany został w C i dystrybuowany jest w postaci źródłowej. Celem zbudowania programu należy wydać komendę `make` w katalogu z źródłami. W przypadku modyfikowania plików nagłówkowych konieczne jest wydanie polecenia `make clean && make`.

Program został podzielony na moduły:

- main.c - główna funkcja programu
- config.c, config.h - obsługa konfiguracji (odczyt i przechowywanie) podawanej w linii poleceń i w pliku konfiguracyjnym
- help.c, help.h - komunikaty pomocy
- signal.c, signal.h - generacja sygnału PCM w oparciu o dane konfiguracyjne
- output_alsa.c, output_alsa.h - odtwarzanie sygnału z wykorzystaniem biblioteki ALSA
- output_dsp.c, output_dsp.h - odtwarzanie sygnału z wykorzystaniem /dev/dsp
- output_uC.c, output_uC.h - odtwarzanie sygnału z wykorzystaniem przystawki elektronicznej

Pobierz[2131]

Generowanie sygnału PCM

Tworzony sygnał PCM przechowywany jest jako tablica 16 bitowych liczb całkowitych ze znakiem. Tablica jest alokowana dynamicznie w oparciu o informacje o częstotliwości próbkowania i długości okresu sygnału. Proste funkcje (kwadrat, sinus, piły i trójkąt) generowane są bezpośrednio w kodzie programu w oparciu o przebieg pętli po kolejnych próbkach należących do okresu sygnału. Bardziej złożone funkcje (tryb `FUNCTION`) generowane są z wykorzystaniem programu calc, który w sposób strumieniowy wykorzystywany jest w pętli przebiegającej po kolejnych próbkach sygnału.

Konwersja do postaci analogowej

Konwersja do postaci analogowej z wykorzystaniem karty dźwiękowej wykorzystuje (do wyboru) mechanizm pliku urządzenia `/dev/dsp` lub bibliotekę ALSA. Realizacja przetwarzania polega na przekazaniu przygotowanej wcześniej tablicy próbek do podsystemu dźwiękowego celem odtworzenia na wyjściu karty dźwiękowej. Metoda ta wiąże się z ograniczeniem możliwych do zastosowania wartości częstotliwości próbkowania do wartości wspieranych przez karty dźwiękowe.

Konwersja do postaci analogowej z wykorzystaniem przystawki mikrokontrolerowej odbywasię się poprzez przekazanie tablicy próbek do mikrokontrolera który odpowiedzialny jest za przekształcenie ich na postać analogową. Próbki przekazywane są z wykorzystaniem portu szeregowego. Ze względu na to iż okres rozsądnego sygnału składa się na ogół z niewięcej niż stu próbek (jest to wynik testów niniejszego programu na karcie dźwiękowej, przy złożonych lub mniej okresowych sygnałach liczba ta może być większe) wszystkie dane przekazywane są za jednym razem. W obecnej wersji przystawka mikrokontrolerowa nie została zrealizowana ze względu na brak w posiadanych zasobach mikrokontrolera z wbudowanym przetwornikiem cyfrowo-analogowym (ku mojemu zaskoczeniu taka funkcjonalność jest żadkością w serii AVR).

SIGNAL.H

```c
#ifndef _SIGNAL_H
#define _SIGNAL_H

#include <inttypes.h>

#include "config.h"

typedef struct signal_t {
 int num_of_sample;
 int16_t *data;
} signal_t;
#define SIGNAL_DATA_MAX 32767
#define ALLOC_SIGNAL_DATA(a) malloc(a.num_of_sample*2)
#define CONVERT_TO_SIGNAL(a) (int16_t)(a*SIGNAL_DATA_MAX)

void run_calc(config_t *config, signal_t *signal);
signal_t signal_prepare(config_t *config);
void signal_free(signal_t *signal);

#endif
```

OUTPUT_uC.H

```c
#ifndef _OUTPUT_UC_H
#define _OUTPUT_UC_H

#define tty_device_speed B115200  // 115200

#include "config.h"
#include "signal.h"

int output_uC_init(config_t *config);

void output_uC_write(int tty_dev, signal_t *signal);

#endif
```

OUTPUT_DSP.H

```c
#ifndef _OUTPUT_DSP_H
#define _OUTPUT_DSP_H

#include "config.h"
#include "signal.h"

int output_dsp_init(config_t *config);
void output_dsp_write(int dsp_dev, signal_t *signal);

#endif
```

OUTPUT_ALSA.H

```
#ifndef _OUTPUT_ALSA_H
#define _OUTPUT_ALSA_H

#include <alsa/asoundlib.h>

#include "config.h"
#include "signal.h"

snd_pcm_t* output_alsa_init(config_t *config);
void output_alsa_write(snd_pcm_t* alsa_dev, signal_t *signal);

#endif
```

HELP.H

```
#ifndef _HELP_H
#define _HELP_H

#include <stdio.h>

void print_help(FILE *output, char *name, int code);
void print_parse_error(int line, const char *info);

#endif
```

CONFIG.H

```
#ifndef _CONFIG_H
#define _CONFIG_H

//#define DEBUG() printf("%s:%d\n", __FILE__, __LINE__);
#define DEBUG()

#include <stdio.h>

typedef enum {
 OUTPUT_DSP = 1,
 OUTPUT_ALSA,
 OUTPUT_UC
} OUTPUT_MODES;

typedef enum {
 SQUARE = 1,
 SIN,
 SAW_R,
 SAW_L,
 TRIANGLE,
 FUNCTION,
 PCM
} MACROS;

typedef struct config_t {
 OUTPUT_MODES output_mode;
 char *output_device_name;
```

```
 int rate;
 FILE *config_file;
 FILE *display_file;

double one_period_time;
 int num_of_periods;

MACROS macro;
 double macro_args[2];

char *function;
 int function_len;
} config_t;

void clean_line(FILE *file);
void clean_white_space(FILE *file);

config_t* config_init(void);
void config_read_cmdline(config_t *config, int argc, char *argv[]);
void config_parse_file(config_t *config);
void config_free(config_t *config);

#endif
```

SIGNAL.C

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <math.h>

#include "signal.h"

void run_calc(config_t *config, signal_t *signal) {
 int write_pipe[2], read_pipe[2];
 if ( pipe(write_pipe) == -1 || pipe(read_pipe) == -1 ) {
  fprintf(stderr, "Error in pipe\n");
  exit(-1);
 }

switch (fork()) {
  case -1:
   fprintf(stderr, "Error in fork\n");
   exit(-1);
  case 0: {
   dup2(write_pipe[0], 0);
   close(write_pipe[1]); close(write_pipe[0]);

   dup2(read_pipe[1], 1);
   close(read_pipe[1]); close(read_pipe[0]);

   execlp("calc", "calc", NULL);
  }

  default: {
   close(write_pipe[0]);
   close(read_pipe[1]);
```

```
   char buf[255], buf_tmp[255];
   double val;
   int i;

   double x = 0;
   double dx = 2 * M_PI / (double)signal->num_of_sample;
   for (i=0; i < signal->num_of_sample; i++){
    sprintf(buf, "x=%f;", x);
    write(write_pipe[1], buf, strlen(buf));
    write(write_pipe[1], config->function, config->function_len);
    read(read_pipe[0], buf, 255);
    sscanf(buf, "%[^0-9]%lf", buf_tmp, &val);
    signal->data[i] = CONVERT_TO_SIGNAL(val);
    buf[20]='\0'; printf ("%c %f -- %s -- %f -- %d\n", buf[0], x, buf, val, signal->data[i]);
    x += dx;
   }
  }
 }
}

signal_t signal_prepare(config_t *config) {
 signal_t signal;
 signal.num_of_sample = config->rate * config->one_period_time; // obliczamy liczbę próbek ⤸
 na jeden okres sygnału
 signal.data = ALLOC_SIGNAL_DATA(signal);

// obliczamy liczbę próbek na jeden okres tworzonego sygnału
 int num_of_nonzero_sample = config->rate * config->macro_args[0];
 if (num_of_nonzero_sample > signal.num_of_sample)
  num_of_nonzero_sample = signal.num_of_sample;
 int i;
 float tmp;

switch(config->macro) {
  case SQUARE: {
   for (i = 0; i < num_of_nonzero_sample; i++) {
    signal.data[i] = SIGNAL_DATA_MAX;
   }
   break;
  }
  case SIN: {
   signal.data[0] = 0;
   double x = 0;
   double dx = 2 * M_PI / (double)num_of_nonzero_sample;

   for (i = 1; i < num_of_nonzero_sample; i++) {
    x += dx;
    signal.data[i] = CONVERT_TO_SIGNAL(sin(x));
   }
   break;
  }
  case SAW_R: {
   int dy = SIGNAL_DATA_MAX / num_of_nonzero_sample;
   signal.data[0] = SIGNAL_DATA_MAX;

   for (i = 1; i < num_of_nonzero_sample; i++) {
    signal.data[i] = signal.data[i-1] - dy;
   }
   break;
```

```
  }
  case SAW_L: {
   int dy = SIGNAL_DATA_MAX / num_of_nonzero_sample;
   signal.data[0] = 0;

   for (i = 1; i < num_of_nonzero_sample; i++) {
    signal.data[i] = signal.data[i-1] + dy;
   }
   break;
  }
  case TRIANGLE: {
   int dy = SIGNAL_DATA_MAX / num_of_nonzero_sample;
   signal.data[0] = 0;

   for (i = 1; i < num_of_nonzero_sample; i++) {
    signal.data[i] = signal.data[i-1] + dy;
   }

   int num_of_nonzero_sample2 = config->rate * config->macro_args[1];
   dy = SIGNAL_DATA_MAX / num_of_nonzero_sample2;
   signal.data[num_of_nonzero_sample] = SIGNAL_DATA_MAX;

   num_of_nonzero_sample2 += num_of_nonzero_sample;
   if (num_of_nonzero_sample2 > signal.num_of_sample)
    num_of_nonzero_sample2 = signal.num_of_sample;
   for (i = num_of_nonzero_sample+1; i < num_of_nonzero_sample2; i++) {
    signal.data[i] = signal.data[i-1] - dy;
   }

   num_of_nonzero_sample = num_of_nonzero_sample2;
   break;
  }
  case FUNCTION: {
   run_calc(config, &signal);
   num_of_nonzero_sample = signal.num_of_sample;
   break;
  }
  case PCM: {
   for (i = 1; i < signal.num_of_sample; i++) {
    if ( fscanf(config->config_file, "%f", &tmp) < 1 ){
     break;
    } else {
     signal.data[i] = CONVERT_TO_SIGNAL(tmp);
     clean_line(config->config_file);
    }
   }
   num_of_nonzero_sample = i;
   break;
  }
 }

// do końca okresu dopełniamy zerami
 for (i = num_of_nonzero_sample; i < signal.num_of_sample; i++) {
  signal.data[i] = 0;
 }

return signal;
}

void signal_free(signal_t *signal) {
```

```
 free(signal->data);
}
```

OUTPUT_UC.C

```
#include "output_uC.h"

#include <stdlib.h>
#include <termios.h>
#include <unistd.h>
#include <fcntl.h>

int output_uC_init(config_t *config) {
 int tty_dev;
 struct termios term;

tty_dev = open (config->output_device_name, O_RDWR);
 if (tty_dev < 0) {
  perror("Opening serial port");
  exit(EXIT_FAILURE);
 }

tcgetattr (tty_dev, &term);
 term.c_iflag = IGNBRK | IGNPAR;
 term.c_oflag = term.c_lflag = term.c_line = 0;
 term.c_cflag = CS8 | CREAD |  CLOCAL | HUPCL | tty_device_speed;
 tcsetattr (tty_dev, TCSAFLUSH, &term);

return tty_dev;
}

void output_uC_write(int tty_dev, signal_t *signal) {
 if ( write(tty_dev, signal->data, 2*signal->num_of_sample) < 2*signal->num_of_sample )
  fputs("error in write to serial device", stderr);
}
```

OUTPUT_DSP.C

```
#include <linux/soundcard.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/ioctl.h>
#include <errno.h>


#include "output_dsp.h"

int output_dsp_init(config_t *config) {
 int dsp_dev, param;

/*if (config->rate != 8000 && config->rate != 16000) {
  fputs("Unsupported bitrate for DSP", stderr);
  exit(EXIT_FAILURE);
 }*/
```

```
dsp_dev = open(config->output_device_name, O_WRONLY);
 if (dsp_dev < 0) {
  perror("Opening DSP device");
  exit(EXIT_FAILURE);
 }

param = config->rate;
 if ( ioctl(dsp_dev, SOUND_PCM_WRITE_RATE, &param) < 0 )
  perror("ioctl on dsp device error");

param = 16;
 if ( ioctl(dsp_dev, SOUND_PCM_WRITE_BITS, &param) < 0 )
  perror("ioctl on dsp device error");

param = 1;
 if ( ioctl(dsp_dev, SOUND_PCM_WRITE_CHANNELS, &param) < 0 )
  perror("ioctl on dsp device error");

return dsp_dev;
}

void output_dsp_write(int dsp_dev, signal_t *signal) {
 if ( write(dsp_dev, signal->data, 2*signal->num_of_sample) < 2*signal->num_of_sample )
  fputs("error in write to dsp", stderr);
}
```

OUTPUT_ALSA.C

```
#include "output_alsa.h"

snd_pcm_t* output_alsa_init(config_t *config) {
 snd_pcm_t *alsa_dev;
 int err;

/*if (config->rate != 8000 && config->rate != 16000) {
  fputs("Unsupported bitrate for ALSA", stderr);
  exit(EXIT_FAILURE);
 }*/

err = snd_pcm_open(&alsa_dev, config->output_device_name, SND_PCM_STREAM_PLAYBACK, 0);
 if (err < 0) {
  printf("ERROR (1b): %s\n", snd_strerror(err));
  exit(EXIT_FAILURE);
 }
 err = snd_pcm_set_params(alsa_dev, SND_PCM_FORMAT_S16, SND_PCM_ACCESS_RW_INTERLEAVED, 1, ⤸
 config->rate, 1, 100000);
 if (err < 0) {
  printf("ERROR (2b): %s\n", snd_strerror(err));
  exit(EXIT_FAILURE);
 }

return alsa_dev;
}

void output_alsa_write(snd_pcm_t* alsa_dev, signal_t *signal) {
 int err;
```

```
err = snd_pcm_writei(alsa_dev, signal->data, signal->num_of_sample);
 if (err < 0) {
  err = snd_pcm_recover(alsa_dev, err, 0);
 }
 if (err < 0) {
  printf("ERROR (3b): %s\n", snd_strerror(err));
  exit(EXIT_FAILURE);
 }
 if (err < signal->num_of_sample) {
  puts("error");
 }
}
```

MAIN.C

```
#include <stdio.h>

#include "config.h"
#include "signal.h"
#include "output_dsp.h"
#include "output_alsa.h"
#include "output_uC.h"

int main(int argc, char *argv[]) {
 int i;

config_t *config;
 config = config_init(); DEBUG()
 config_read_cmdline(config, argc, argv); DEBUG()
 config_parse_file(config); DEBUG()

printf("# OUTPUT_MODES=%d rate=%d one_period_time=%f num_of_periods=%d\n",
  config->output_mode, config->rate, config->one_period_time, config->num_of_periods);
 printf("# MACRO=%d (%f, %f)\n",
  config->macro, config->macro_args[0], config->macro_args[1]);
 printf("# FUNCTIONS STRING:\n# \t%s\n",
  config->function);

/// utworzenie pojedynczego okresu sygnalu w oparciu o konfig;
 signal_t signal = signal_prepare(config); DEBUG()

/// opcjonalny dump sygnału do wskazanego pliku
 if (config->display_file != NULL) {
  for(i=0; i<signal.num_of_sample; i++)
   fprintf(config->display_file, "%d\n", signal.data[i]);
  fclose(config->display_file);
  config->display_file = NULL;
 }
 DEBUG()



/// wysyłanie sygnału na wyjście audio lub do uC

if (config->output_mode == OUTPUT_DSP) {
  /// DSP
```

```
  DEBUG()
  int dsp_dev = output_dsp_init(config);

  if (config->num_of_periods > 0)
   for(i=0; i<config->num_of_periods; i++)
    output_dsp_write(dsp_dev, &signal);
  else
   while(1)
    output_dsp_write(dsp_dev, &signal);

  close(dsp_dev);
 } else if (config->output_mode == OUTPUT_ALSA) {
  /// ALSA
  DEBUG()
  snd_pcm_t *alsa_dev = output_alsa_init(config);

  if (config->num_of_periods > 0)
   for(i=0; i<config->num_of_periods; i++)
    output_alsa_write(alsa_dev, &signal);
  else
   while(1)
    output_alsa_write(alsa_dev, &signal);

  snd_pcm_close(alsa_dev);
 } else if (config->output_mode == OUTPUT_UC) {
  /// uC czyli port szeregowy
  DEBUG()
  int uC_dev = output_uC_init(config);
  output_uC_write(uC_dev, &signal);
 } else {
  fputs("Unknow output mode", stderr);
 }
 DEBUG()

signal_free(&signal); DEBUG()
 config_free(config); DEBUG()

return 0;
}
```

HELP.C

```
#include "help.h"

#include <stdlib.h>
#include <stdarg.h>

void print_help(FILE *output, char *name, int code) {
 fprintf(output, "USAGE:\n");
 fprintf(output, "%s -o dsp|alsa|uC -r val [-d disp_file] config_file\n", name);
 fprintf(output, "  -o dsp|alsa|uC     Output mode\n");
 fprintf(output, "  -O device          Output device\n");
 fprintf(output, "  -r val             sampling Rate in Hz\n");
 fprintf(output, "  -d disp_file       write data to Display file\n");
 fprintf(output, "  config_file is a text file contains signal description\n");
 fprintf(output, "\n");
 fprintf(output, "Format of config_file:\n");
 fprintf(output, "TIME number_of_periods time_of_one_period\n");
```

```
 fprintf(output, "MACRO_NAME macro_parametr\n");
 exit(code);
}

void print_parse_error(int line, const char *info) {
 fprintf(stderr, "ERROR: Unparsable line %d in config file\n", line);
 if (info != NULL) fputs(info, stderr);
 exit(-1);
}
```

CONFIG.C

```
#define _GNU_SOURCE

#include "config.h"
#include "help.h"

#include <getopt.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>

float to_seconds(float val, const char *unit) {
 if (unit == NULL || unit[0] == '\0') {
  val = val;
 } else if (unit[0] == 'm') {
  val = val / 1000.0;
 } else if (unit[0] == 'u') {
  val = val / 1000000.0;
 } else {
  val = NAN;
 }
 return val;
}

void clean_line(FILE *file) {
 char buf[20];
 while (fscanf(file, "%20[^\n]", buf) > 0);
}
void clean_white_space(FILE *file) {
 char buf[20];
 while (fscanf(file, "%[ \t]", buf) > 0);
}

config_t* config_init(void) {
 return (config_t *) calloc(1, sizeof(config_t));
}

void config_read_cmdline(config_t *config, int argc, char *argv[]) {
 int opcja;
 opterr = 0;
 while((opcja = getopt(argc, argv, "o:O:r:d:h")) != -1) {
  switch (opcja) {
   case 'o':
    if (strcmp("dsp", optarg) == 0) {
     config->output_mode = OUTPUT_DSP;
    } else if (strcmp("alsa", optarg) == 0) {
```

```
      config->output_mode = OUTPUT_ALSA;
    } else if (strcmp("uC", optarg) == 0) {
      config->output_mode = OUTPUT_UC;
    }
    break;
  case 'O':
    config->output_device_name = malloc(strlen(optarg));
    stpcpy(config->output_device_name, optarg);
    break;
  case 'r':
    config->rate = atoi(optarg);
    break;
  case 'd':
    if (strcmp("-", optarg) == 0) {
      config->display_file = stdout;
    } else {
      config->display_file = fopen(optarg, "w+");
      if (config->display_file == NULL) {
        fprintf(stderr, "ERROR: Can't open %s file\n", optarg);
        exit(-1);
      }
    }
    break;
  case 'h':
    print_help(stdout, argv[0], 0);
  default:
    print_help(stderr, argv[0], -1);
  }
 }

if (config->output_mode == 0 || config->rate == 0) {
  fprintf(stderr, "ERROR: Incorect -o or -r value\n");
  print_help(stderr, argv[0], -1);
 }

if (optind < argc) {
  config->config_file = fopen(argv[optind], "r");
  if (config->config_file == NULL) {
   fprintf(stderr, "ERROR: Can't open %s file\n", argv[optind]);
   exit(-1);
  }
 } else {
  print_help(stderr, argv[0], -1);
 }
}

void config_parse_file(config_t *config) {
 char buf[1024], bufU[3];
 double bufV;
 int code, num, num_base;

/// FIRST LINE
 bufU[0] = '\0';
 code = fscanf(
  config->config_file,
  "%20s %d %lf%[^0-9. \t\n]2s",
  buf,
  &(config->num_of_periods),
  &bufV,
  bufU
```

```
);

if (code < 3 || strcmp("TIME", buf) != 0) print_parse_error(1, NULL);

config->one_period_time = to_seconds(bufV, bufU);
 if ( isnan(config->one_period_time) ) {
  print_parse_error(1, "Unknown SI prefix\n");
 }

clean_line(config->config_file);


/// SECOND LINE
 code = fscanf(config->config_file, "%20s", buf);
 if (code != 1) print_parse_error(2, "No macro name?\n");

if (strcmp("SQUARE", buf) == 0) {
  config->macro = SQUARE;
  num = 1;
 } else if (strcmp("SIN", buf) == 0) {
  config->macro = SIN;
  num = 1;
 } else if (strcmp("SAW_R", buf) == 0) {
  config->macro = SAW_R;
  num = 1;
 } else if (strcmp("SAW_L", buf) == 0) {
  config->macro = SAW_L;
  num = 1;
 } else if (strcmp("TRIANGLE", buf) == 0) {
  config->macro = TRIANGLE;
  num = 2;
 } else if (strcmp("FUNCTION", buf) == 0) {
  config->macro = FUNCTION;
  num = -1;
 } else if (strcmp("PCM", buf) == 0) {
  config->macro = PCM;
  num = 0;
 } else {
  print_parse_error(2, "Unknown macro name\n");
 }

num_base = num - 1;

if (num < 0) {
  clean_white_space(config->config_file);
  code = fscanf(
   config->config_file,
   "%1023c",
   buf
  );
  if (code < 1) {
   print_parse_error(2, "Error parsing function string\n");
  }

  buf[1023] = '\0';
  char *tmp = index(buf, '\n');
  if (tmp == NULL) {
   print_parse_error(2, "too long or no new line at end function string\n");
  } else {
   // zaraz po znaku nowej linii kończymy napis
```

```
   *(tmp+1) = '\0';
  }

  config->function_len = (int)(tmp - buf) + 2;
  config->function = malloc(config->function_len);
  strncpy(config->function, buf, config->function_len);
 } else while (num-- > 0) {
  bufU[0] = '\0';

  code = fscanf(
   config->config_file,
   "%lf%[^0-9. \t\n]2s",
   &(config->macro_args[num_base-num]),
   bufU
  );

  if (code < 1) {
   if (num_base == 0) {
    fprintf(stderr, "WARNING: Use time_of_one_period instand of macro %d parametr\n", num);
    config->macro_args[0] = config->one_period_time;
    continue;
   } else {
    print_parse_error(2, "No macro parametr?\n");
   }
  }

  config->macro_args[num_base-num] = to_seconds(config->macro_args[num_base-num], bufU);
  if ( isnan(config->macro_args[num_base-num]) ) {
   print_parse_error(2, "Unknown SI prefix\n");
  }
 }

clean_line(config->config_file);
}


void config_free(config_t *config) {
 fclose(config->config_file);
 if (config->function != NULL)
  free(config->function);
 if (config->output_device_name != NULL)
  free(config->output_device_name);
 free(config);
}
```

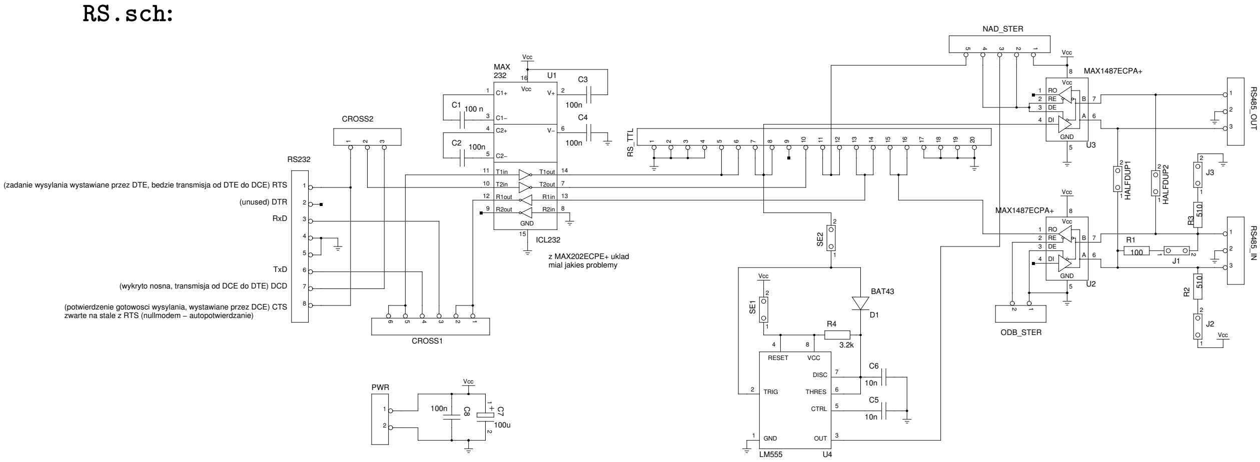
Konwerter pomiędzy różnymi standardami UART (RS232, RS485, RS_TTL)

Przedstawiam projekt konwertera pomiędzy różnymi standardami portu szeregowego (UART[2132]). Podstawowym poziomem pośrednim w tym układzie jest TTL 5V (jest on także wyprowadzony jako jeden z wariantów portu). Układ z LM555 służy do sterowania nadajnikiem RS485, alternatywnie może być sterowany z sygnałów DCD/RTS. RS-485[2133]/RS-422[2134] może pracować w trybie half-duplex i full-duplex (ustawiane zworkami), w przypadku full-duplexu, odbiornik powinien być zawsze załączony a nadajnik może być załączony stale lub sterowany tak jak przy half-duplexie.

Zamieszczam schemat ideowy[2135](PDF: http://www.opcode.eu.org//usage_and_config/

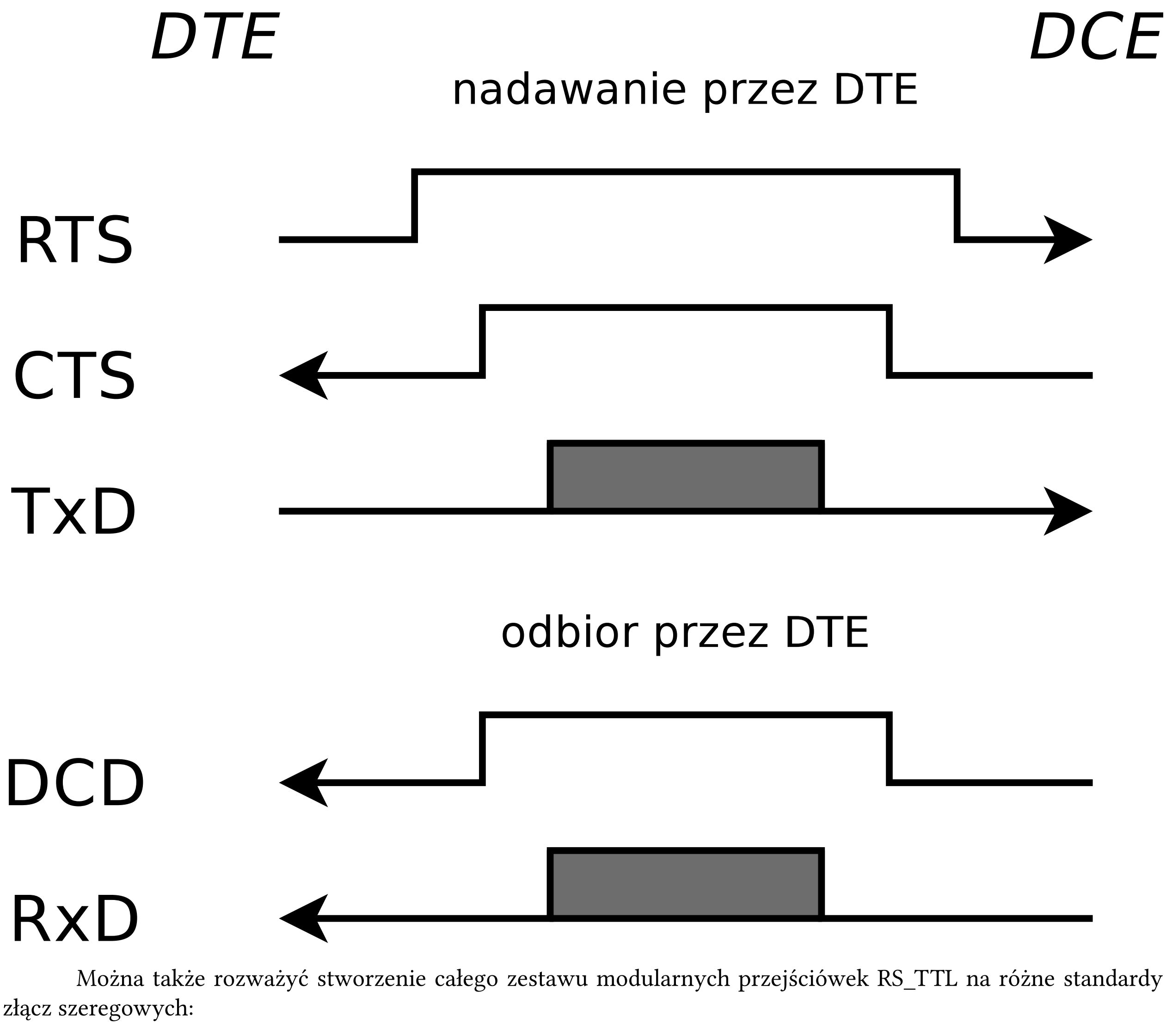
toolbox4electronic/konwerter_RS/RS.pdf), projekt płytki[2136](PDF: http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_RS/RS.pcb.pdf).

RS.sch:



Dostępny jest także plik pokazujący mapowanie pinów RS-232[2137] w zastosowanym złączu RJ45 na standardowe DB-9 - RS232-via-RJ45.svg[2138], zobacz też Yost Serial Device Wiring Standard[2139].

Zamieszczam także wykres przedstawiający zależności w wystawianiu sygnałów sterujących transmisją RS232:



Można także rozważyć stworzenie całego zestawu modularnych przejściówek RS_TTL na różne standardy złącz szeregowych:

```
zlacze (RJ-45 albo goldpin)
---------------------------
numer  kierunek   opis
    (net.    mod.)
```

```
  1       ->      vcc_in
  2       ->      rx_en
  3      <->      gnd
  4       ->      rx
  5      <-       tx
  6      <->      gnd
  7      <-       tx_en
  8      <-       vcc_out
+---------+       +---------+
| 1 2 3 4 |       | 8 7 6 5 |
| 5 6 7 8 |  <->  | 4 3 2 1 |
+---------+       +---------+
moduly
------
1) rs_ttl <-- usb (jako adapter do posiadanej przejsciówki)
2) rs_ttl <-> transoptory <-> rs_ttl
3) rs_ttl <-> rs_485 (full/half)
4) rs_ttl <-> rs_232
5) rs_ttl --> mikrokontroler <-> data_bus/i2c/spi
                             <-> i2c <-> DS2482 <-> 1wire
                             <-> spi <-> MCP2515 <-> CAN
6) rs_ttl <-> DS2480B <-> 1wire
7) rs_ttl <-> MCP2003-E/P <-> LIN
```

Zobacz w Sieci: Klocki RS-485, część 1 (kit AVT-530)[2140]

Konwerter poziomów TTL

Dominującymi standardami napięć w współczesnej elektronice cyfrowej jest 5V i 3.3V. Istotnym zagadnieniem staje się często konwersja poziomów sygnałów pomiędzy tymi standardami. Zagadnienie to można rozwiązać na kilka sposobów:

— z wykorzystaniem buforów jednokierunkowych (np. 74HC125, szczegóły poniżej)
— z wykorzystaniem bufora dwukierunkowego (np. 74HC245) - rozwiązanie bardzo podobne do poprzedniego (jako że w zasadzie są to naprzemiennie działające bufory trójstanowe w przekazujące w przeciwne strony) ale wygodne przy magistralach dwukierunkowych (szynach danych)
— poprzez zastosowanie optoizolatorów, dodatkową zaletą takiego rozwiązania jest zapewnienie izolacji galwanicznej pomiędzy sygnałami (schemat ideowy[2141] oraz projekt płytki[2142])

Przedstawiam także projekt prostego konwertera poziomów TTL 5V i 3.3V opartego na dwóch wspomnianych układach 74125 (AHCT i AHC[2143] - porównaj w dokumentacji charakterystyki DC dla tych rodzin).

Zasada działa układu opiera się na tym iż układ AHCT zasilany z 5V otrzymując na wejściu sygnał w standardzie CMOS 3V (LVTTL) produkuje na wyjściu sygnał w standardzie 5V TTL. Natomiast układ AHC zasilany z 3V toleruje i poprawnie interpretuje sygnały 5V TTL dając na wyjściu sygnały logiki 3.0V. Dodatkowe złącza pinowe (J1-J8) umożliwiają sterowanie sygnałami OE (np. celem uzyskania konwertera dwukierunkowego), w przypadku gdy nie mamy tego robić zworka powinna być w pozycji 1-2 (pin OE podłączony do GND).

Należy wspomnieć iż stan wysoki w TTL to napięcie z zakresu 2-5V, niski 0-0.8V. Natomiast w przypadku CMOS (tak wykonywana jest większość współczesnych układów ”TTL”) wynoszą one odpowiednio VCC/2-VCC i 0-VCC/2.

projekt w trakcie testów - coś może nie działać, mogą nastąpić istotne zmiany

Zamieszczam schemat ideowy[2144], projekt płytki[2145]

Konwerter pomiędzy USB a UART (TTL)

Przedstawiam projekt konwertera pomiędzy USB a UART. Bardzo podobny układ znajdziemy w kablach USB-RS232 (ale dający poziomy RS232 a nie TTL), jak również w kablach USB do starszych telefonów (mających w złączu tak naprawdę UART, a nie USB). Ze względów finansowych bardziej opłacalne jest wykorzystanie kabla do telefonu niż budowa takiego układu.

Zamieszczam schemat ideowy[2146], projekt płytki[2147], plik projektu xgsch2pcb[2148].

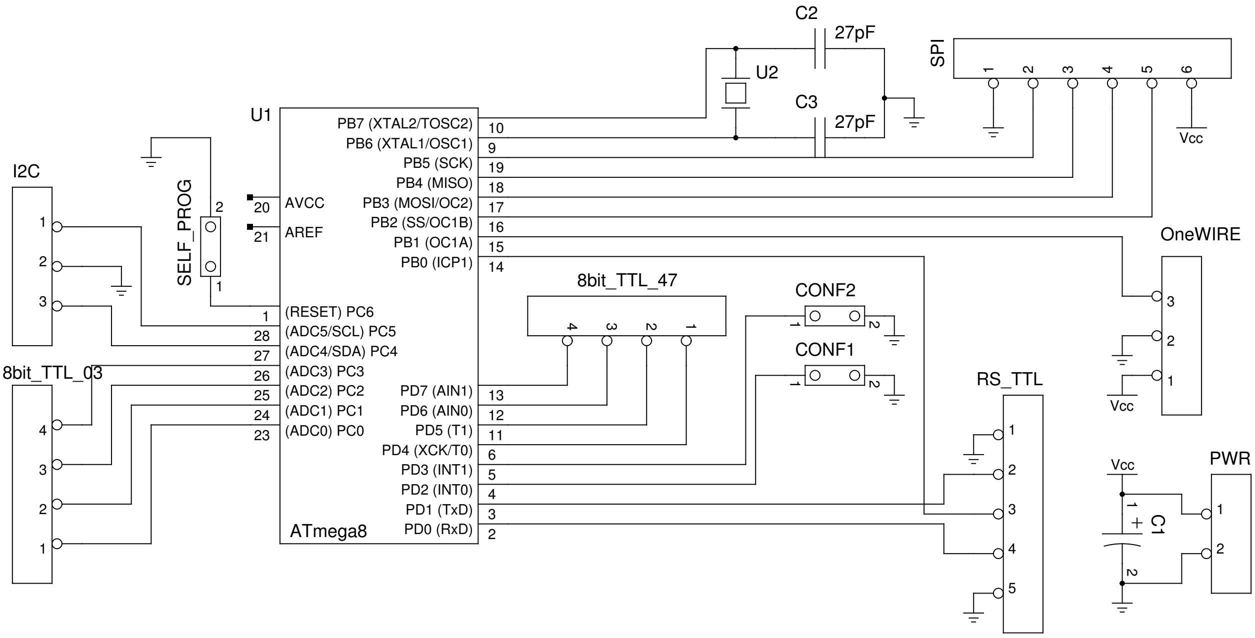
Konwerter RS do 8bit/OneWire/I2C/SPI

Jest to projekt przejściówki z RS na różne przydatne w elektronice interfejsy a konkretnie: 8bit, 1-Wire[2149], I2C[2150], Serial Peripheral Interface[2151]. Za konwersję odpowiada mikrokontroler Atmega8.

projekt w trakcie realizacji - nie był jeszcze wykonany prototyp ani testy, coś może nie działać, mogą nastąpić istotne zmiany; program sterujący nie jest dostosowany do obecnej wersji przejściówki.

- schemat ideowy[2152](PDF: `http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter-interfejsy_elektroniczne/rs-i2c_spi_1wire_8bit.pdf`)
- projekt PCB[2153](PDF: `http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter-interfejsy_elektroniczne/rs-i2c_spi_1wire_8bit.pcb.pdf`)
- program sterujący praca mikrokontrolera (wymagane nagłówki: onewire.c i opoznienia.c)

`rs-i2c_spi_1wire_8bit.sch:`



Wstępnie przygotowany jest też zalążek wersji z obsługą ethernetu (w chwili obecnej tylko układ mikrokontrolera w wersji Atmega32): ideowy[2154] i projekt PCB[2155]. Układ ten będzie komunikował się z ethernetem za pomocą kontrolera MAC i PHY - "ENC28J60". Więcej na temat podłączania tego układu do AVR i TCP/IP na Atmega32 w Serwer sieciowy na AVR[2156] (strona domowa projektu etherrape[2157]) oraz w ETHERNET - "karta sieciowa" SPI dla AVR enc28j60[2158].

PRZEJSCIOWKA.C

```
/**
 * program dla mikrokontrolera Atmega8 stanowiącego interfejs
 * pomiędzy portem szeregowym RS, a 1wire, I2C, SPI, 8bit (równoległey)
 *
 * informacje z rs odbierane są w trybie przerwaniowym,
 * po odebraniu informacji sprawdzany jest stan PB0 (zworka CONF1) ->
 * jeżeli jest zwarta do GND następuje odczytanie informacji o trybie pracy
 * i wykonanie odpowiedniej akcji konfiguracyjnej, obsługiwane tryby
 *    0x01 -> SPI (dane z RS wysyłane na SPI, dane z SPI odbierane przez przerwanie i ⤸
 wysyłane do RS)
 *    0x02 -> I2C/TWI Master (jak wyżej)
 *    0x03 -> 1 wire (komunikat RS składa się w ogólności z kilku bajtów:
 *       pierwszy bajt informuje o trybie pracy 1wire {bit 0 = wykonywać reset, bit 1 = ⤸
 wysyłać
```

```
*       pominięcie ROM /w przeciwnym razie oczekiwany i wysyłany ROM/ , bit 2 = wysłać 0xff,
*       bit 3 = 1 /musi być ustawiony gdyby inaczej bajt miał wartość 0/, bity 4-7 = ⤸
ilość bajtów komendy}
*       następnie w zależności od wartości pierwszego bajtu następują kolejno: adres ROM (⤸
8 bajtów)
*       i ciąg bajtów komendy {należy zaznaczyć że nie wszystkie ustawienia flag ⤸
pierwszego bajtu mają sens,
*       w szczególności ustawienie bitu 2 wymaga wyzerowania wszystkich pozostałych,
*       wysłanie takiego pakietu powoduje odczyt 1 bajtu danych z 1-wire i wysłanie go na RS
*       rozpoczęcie transmisji wymaga ustawienia bitu zerowego, może być także ustawiony ⤸
bit pierwszy}
*   0x04 -> 8bit OUT (zapis do RS powoduje wysłanie danych na wyjście równoległe)
*   0x05 -> 8bit IN (zapis do RS powoduje odczyt danych z wejścia równoległego i wysłanie ⤸
ich na RS)
*   0x06 -> 8bit IN-OUT 1/2 (dla bitów 0..3 IN dla bitów 4..7 OUT)
*   0x07 -> 8bit IN-OUT FULL (zapis do RS powoduje odczyt z równoległych,
*       zapis nowej wartości na równoległe i wysłanie odczytanej na RS)
*   0x08 -> 8bit PRZERWANIOWY (przerwanie INT0 powoduje odczyt danych z linii równoległych ⤸
i wysłanie na RS)
*   @ wszystkie tryby 8bit -> zapis na linie równoległe powoduje zmianę stanu linii INT1
*
*
* kompilacja i wgranie przez ISP:
*   avr-gcc -mmcu=atmega8 -Os -o sterownik.o sterownik.c
*   avr-objcopy -O ihex sterownik.o sterownik.hex
*   uisp -dprog=dapa -dlpt=0x378 --erase --segment=flash
*   uisp -dprog=dapa -dlpt=0x378 --upload --segment=flash if=sterownik.hex
*   uisp -dprog=dapa -dlpt=0x378 --verify --segment=flash if=sterownik.hex
*   w razie problemow warto przeladowac modul odpowiedzialny za obsluge LPT
*       rmmod lp parport_pc parport ; modprobe parport_pc
*
* konfiguracja układu na zewnętrzny kwarc 3MHz
*   uisp -dprog=dapa -dlpt=0x378 --wr_fuse_l=0xEF
*/


#include <inttypes.h>

#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>

#define F_CPU 1000000UL
#include <util/delay.h>

/** nagłówki OneWire **/
#define onewire_send_zero()   DDRB |= 0x02; PORTB &= 0xfd
#define onewire_send_jeden()  PORTB |= 0x02; DDRB &= 0xfd
#define onewire_read()        (PINB & 0x02) >> 6
#include "onewire.c"

/// tryb w którym prcuje układ
volatile uint8_t config_mode;

/// licznik timera programowego
volatile int8_t soft_timer;

/** obsluga portu szeregowego **/
uint8_t uart_buf_head, uart_buf_body, chanel, cmd = 0;
ISR(USART_RXC_vect) {
```

```
cli();
uint8_t tmp, dane;
uint8_t rom[8];
uint8_t rom_num;
if (PINC & 0x01) { /// TRYB KONFIGURACYJNY
 // zapamietanie identyfikatora trybu
 config_mode = UDR;

 // wyczyszczenie wszystkich wczesniejszych ustawień
 TCCR0 = 0;
 GIMSK = 0; MCUCR = 0;
 DDRB = 0; PORTB = 0x01;
 DDRC = 0; PORTC = 0;
 DDRD = 0; PORTD = 0;
 TWCR = 0;

 // wprowadzenie nowych ustawień
 switch config_mode {
  case 0x01:
   // podajemy RESET dla ISP, MOSI i SCK na output
   DDRB=0x2c; PORTB=0x01;
   // właczenie SPI jako master z zegarem = 1/128 zegara własnego i generowaniem przerwań
   SPCR = (1<<SPE) | (1<<MSTR) | (1<<SPR0) | (1<<SPR1) | (1<<SPIE);
   //FIXME jakie ustawienia poniższych parametrów do programowania przez ISP ?
   //DORD - kolejnośc transmisji bitów w bajcie
   //CPOL - który stan zegara aktywny a który nie
   //CPHA - przy którym zboczu odbiór danych
   break;
  case 0x02:
   tmp = 0; // stan transmisji - nierozpoczęta
   PORTC = 0x30; // rezystory podciągające
   TWSR = 0xfa; // preskaler == 16
   TWBR = 100; // = CPU_clock / (SLC_clock*2*4^16) - 16, >10
   TIMSK = _BV(TOIE0); // przerwanie zegarowe
   break;
  case 0x03:
   tmp = 0;
   break;
  case 0x04:
   DDRC = 0x0f; PORTC = 0;
   DDRD = 0xf8; PORTD = 0;
   tmp = 0x00;
   break;
  case 0x05:
   DDRC = 0; PORTC = 0x0f;
   DDRD = 0; PORTD = 0xf8;
   break;
  case 0x06:
   DDRC = 0; PORTC = 0x0f;
   DDRD = 0xf8; PORTD = 0;
   tmp = 0x00;
   break;
  case 0x08:
   // wyjściowy (wejście z INT0 == PD2)
   DDRC = 0x0f; PORTC = 0;
   DDRD = 0xf8; PORTD = 0;
   // przerwania z int0
   GIMSK = (1 << INT0);
   MCUCR = (1 << ISC00) | (1 << ISC01);
   break;
```

```
 }
} else { /// TRYB PRACY
 dane = UDR;
 switch config_mode {
  case 0x01:
   SPDR = dane;
   // czekanie na koniec
   while(!(SPSR & (1<<SPIF)));
   break;
  case 0x02:
   /// START
   if (tmp == 0) {
    // zaczynamy, master, włączony, przerwanie
    TWCR = (1<<TWINT) | (1<<TWSTA) | (1<<TWEN) | (1<<TWIE) | (1<<TWEA);
    // czekamy na koniec inicjalizacji
    while(!(TWCR & (1<<TWINT)));
   }
   /// normalna transmisja
   TWDR = dane;
   TWCR = (1<<TWINT) | (1<<TWEN) | (1<<TWIE) | (1<<TWEA);
   while (!(TWCR & (1<<TWINT)));
   /// STOP w oparciu o przerwanie z TIMER 10s
   TCNT0 = 0; // ustawienie wartości początkowej TIMER0
   soft_timer = 120;
   TCCR0 = 0x05; // wzrost wartości TIMER0 co ... taktów zegara głównego (5 => 1024)
   break;
  case 0x03:
   if (tmp==0) {
    tmp = dane;
    soft_timer = (tmp >> 4) & 0x0f;
    rom_num = 0;
   } else if (tmp & 0x01) { // wykonujemy reset
    if (tmp & 0x02) { // wysyłamy ROM
     if(rom_num < 8) {
      rom[rom_num++] = dane;
     } else {
      onewire_command(dane, rom);
      while ( !onewire_bit(1) );
      soft_timer--;
     }
    } else {
     onewire_command(dane, NULL);
     while ( !onewire_bit(1) );
     soft_timer--;
    }
   } else if (tmp & 0x04) { // nie wykonujemy resetu, odczyt
    UDR = onewire_reply();
    tmp = 0;
   } else { // nie wykonujemy resetu
    onewire_byte(dane);
    soft_timer--;
   }
   if (soft_timer == 0)
    tmp = 0;
   break;
  case 0x07:
   DDRC = 0; PORTC = 0x0f;
   DDRD = 0; PORTD = 0xf8;

   _delay_us(2);
```

```
    UDR = (PINC & 0x0f) | (PIND & 0xf0);

    DDRC = 0x0f; DDRD = 0xf8;
    // nie przerywamy - trzeba jesze ustawić nową wartość
   case 0x04:
   case 0x08:
    tmp = ~tmp;
    PORTC = dane & 0x0f;
    PORTD = (dane & 0xf0) | (tmp & 0x08);
    break;
   case 0x06:
    tmp = ~tmp;
    UDR = (PINC & 0x0f);
    PORTD = (dane & 0xf0) | (tmp & 0x08);
    break;
   case 0x05:
    UDR = (PINC & 0x0f) | (PIND & 0xf0);
    break;
  }
 }
 sei();
}

/** obsluga przerwania z SPI/ISP **/
SIGNAL(SIG_SPI) {
 UDR = SPDR;
}

/** obsluga przerwania z I2C/TWI **/
ISR(TWI_vect) {
 UDR = TWDR;
 // wyczyszenie TWINT
 TWCR = (1<<TWINT) | (1<<TWEN) | (1<<TWIE) | (1<<TWEA);
}

/** obsluga przerwania z INT0 **/
SIGNAL (SIG_INTERRUPT0) {
 uint8_t c, d;
 c = PORTC; d = PORTD;
 DDRC = 0; PORTC = 0x0f;
 DDRD = 0; PORTD = 0xf8;

 _delay_us(2);
 UDR = (PINC & 0x0f) | (PIND & 0xf0);

DDRC = 0x0f; PORTC = c;
 DDRD = 0xf8; PORTD = d;
}

/** obsługa przerwania z TIMER0 **/
SIGNAL (SIG_OVERFLOW0) {
 if (--soft_timer == 0) {
  /// wyłączamy TIMER0
  TCCR0 = 0;

  /// sygnał STOP dla I2C/TWI
  TWCR = (1<<TWINT) | (1<<TWEN) | (1<<TWSTO) | (1<<TWIE);
  while ((TWCR & (1<<TWSTO)));
 }
}
```

```
main() {
  /** port szeregowy - inicjalizacja **/
/// włączenie nadajnika, odbionika UART wraz z generowaniem przerwań
 UCSRB = (1 << TXEN) | (1 << RXEN) | (1 << RXCIE);
 /// ustawienie prędkości (BAUD), UBRR = F_CPU / (16*BAUD) -1
 UBRRH = 0; UBRRL = 19; // BAUD = 9600 przy 3 MHz
 //UBRRH = 0; UBRRL = 207; // BAUD = 300 przy 1 MHz

/** włączenie przerwań **/
sei();

/** konfiguracja portów IO mikrokontrolera **/
/// PB0 - zworka konfiguracyjna
 DDRB = 0;
 PORTB = 0x01;

/// PB1 - One Wire
 /// PC0-PC3, PD4-PD7 - 8 bit IO
 /// PD2, PD3 - zworki dodatkowe
 /// Konfigurowane w zależności od trybu pracy ...

/** petla glowna **/
while (1);
}
```

ERROR_REPORTING.H

```
#ifndef ERROR_REPORTING_H
#define ERROR_REPORTING_H

#ifdef NO_USE_SYSLOG
#include <err.h>
# define LOG_PRINT_DEBUG(txt, ...)   warnx("DEBUG: "     txt " (at %s:%d)", ##__VA_ARGS__, ↩
__FILE__, __LINE__)
# define LOG_PRINT_NOTICE(txt, ...)  warnx("NOTICE"      txt " (at %s:%d)", ##__VA_ARGS__, ↩
__FILE__, __LINE__)
# define LOG_PRINT_INFO(txt, ...)    warnx("INFO: "      txt " (at %s:%d)", ##__VA_ARGS__, ↩
__FILE__, __LINE__)
# define LOG_PRINT_WARN(txt, ...)    warnx("WARNING: "   txt " (at %s:%d)", ##__VA_ARGS__, ↩
__FILE__, __LINE__)
# define LOG_PRINT_ERROR(txt, ...)   warnx("ERROR: "     txt " (at %s:%d)", ##__VA_ARGS__, ↩
__FILE__, __LINE__)
# define LOG_PRINT_CRIT(txt, ...)    warnx("CRITICAL: "  txt " (at %s:%d)", ##__VA_ARGS__, ↩
__FILE__, __LINE__)
#else
# include <syslog.h>
# define LOG_PRINT_DEBUG(...)   syslog(LOG_DEBUG,    __VA_ARGS__)
# define LOG_PRINT_NOTICE(...)  syslog(LOG_NOTICE,   __VA_ARGS__)
# define LOG_PRINT_INFO(...)    syslog(LOG_INFO,     __VA_ARGS__)
# define LOG_PRINT_WARN(...)    syslog(LOG_WARNING,  __VA_ARGS__)
# define LOG_PRINT_ERROR(...)   syslog(LOG_ERR,      __VA_ARGS__)
# define LOG_PRINT_CRIT(...)    syslog(LOG_CRIT,     __VA_ARGS__)
#endif

#endif
```

ZABBIX.C

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>

#include <sys/socket.h>
#include <netinet/ip.h>

#include "error_reporting.h"
#include "zabbix.h"

void zabbix_message_reinit(zabbix_msg_t *msg) {
 msg->size = 0;
 msg->is_item = 0;
 msg->size += snprintf(msg->data + msg->size, msg->max_size - msg->size, "{\"request\":⤸
 \"sender data\",\"data\":[\n");
}

void zabbix_message_add_item(zabbix_msg_t *msg, const char *host, const char *item, const ⤸
char *value) {
 char extra[3];
 if (msg->is_item) {
  extra[0] = ',';
  extra[1] = '\n';
  extra[2] = 0;
 } else {
  extra[0] = 0;
  msg->is_item = 1;
 }
 msg->size += snprintf(msg->data + msg->size, msg->max_size - msg->size,
  "%s\t{\"host\":\"%s\",\"key\":\"%s\",\"value\":\"%s\"}", extra, host, item, value);
}

void zabbix_message_add_item_t(zabbix_msg_t *msg, const char *host, const char *item, const ⤸
char *value, int time) {
 char extra[3];
 if (msg->is_item) {
  extra[0] = ',';
  extra[1] = '\n';
  extra[2] = 0;
 } else {
  extra[0] = 0;
  msg->is_item = 1;
 }
 msg->size += snprintf(msg->data + msg->size, msg->max_size - msg->size,
  "%s\t{\"host\":\"%s\",\"key\":\"%s\",\"value\":\"%s\",\"clock\":%d}", extra, host, item, ⤸
  value, time);
}

void zabbix_message_add_item_f(zabbix_msg_t *msg, const char *host, const char *item, float ⤸
value) {
 char extra[3];
 if (msg->is_item) {
  extra[0] = ',';
  extra[1] = '\n';
  extra[2] = 0;
 } else {
  extra[0] = 0;
```

```
  msg->is_item = 1;
 }
 msg->size += snprintf(msg->data + msg->size, msg->max_size - msg->size,
  "%s\t{\"host\":\"%s\",\"key\":\"%s\",\"value\":\"%f\"}", extra, host, item, value);
}

void zabbix_message_add_item_tf(zabbix_msg_t *msg, const char *host, const char *item, float ⤸
value, int time) {
 char extra[3];
 if (msg->is_item) {
  extra[0] = ',';
  extra[1] = '\n';
  extra[2] = 0;
 } else {
  extra[0] = 0;
  msg->is_item = 1;
 }
 msg->size += snprintf(msg->data + msg->size, msg->max_size - msg->size,
  "%s\t{\"host\":\"%s\",\"key\":\"%s\",\"value\":\"%f\",\"clock\":%d}", extra, host, item, ⤸
  value, time);
}

int zabbix_message_finish(zabbix_msg_t *msg) {
 msg->size += snprintf(msg->data + msg->size, msg->max_size - msg->size, "]}");
 //msg->size += snprintf(msg->data + msg->size, msg->max_size - msg->size, "],\"clock\":%d}", ⤸
 time(0));
 if (msg->size > 0xfff0)
  return -1;
 else
  return 0;
}

int zabbix_write(int fd, zabbix_msg_t *msg) {
 int a;
 char header[13];

/// prepare header
 header[0]='Z'; header[1]='B'; header[2]='X'; header[3]='D'; header[4]=0x01;
 header[5] = msg->size % 256;
 header[6] = msg->size / 256;
 for (a=7; a<13; a++) header[a] = 0;

/// write data
 a = write(fd, header, 13);
 if (a != 13) {
  return -1;
 }
 a = write(fd, msg->data, msg->size);
 if (a != msg->size) {
  return -1;
 }

return 0;
}

int zabbix_listen(int *sh) {
 char buf[100];
 struct timeval timeout;
 fd_set zabbix_fd_set;
```

```
usleep(500000);

FD_ZERO(&zabbix_fd_set);
 FD_SET(sh[0], &zabbix_fd_set);

timeout.tv_sec = 0;
 timeout.tv_usec = 500000;

int ret = select(sh[0]+1, &zabbix_fd_set, NULL, NULL, &timeout);
 if (ret > 0) {
  ret=read(sh[0], buf, 100);
#ifdef DEBUG
  if (ret > 0) {
   buf[ret]=0;
   int i;
   for (i=13; i<ret; i++)
    if (buf[i]<0x20)
     buf[i]=' ';
   LOG_PRINT_DEBUG("%s", buf+13);
  }
#endif
 } else if (sh[1] == 1) {
  LOG_PRINT_WARN("zabbix_listen - timeout or error in select()");
 }

close(sh[0]);
 free(sh);

return 0;
}

int zabbix_send(zabbix_msg_t *msg) {
 if ( zabbix_message_finish(msg) < 0 ) {
  LOG_PRINT_WARN("zabbix_send - too big message");
  return -1;
 }

int *sh = malloc(sizeof(int));
 sh[0] = socket(PF_INET, SOCK_STREAM, 0);
 if (*sh<0) {
  LOG_PRINT_WARN("zabbix_send - socket: %m");
  return -2;
 }

if ( connect(sh[0], (struct sockaddr *) msg->zabbix_server, sizeof(struct sockaddr_in)) ) {
  LOG_PRINT_WARN("zabbix_send - connect: %m");
  close(sh[0]);
  return -3;
 }

if (zabbix_write(sh[0], msg) < 0){
  LOG_PRINT_WARN("zabbix_send - zabbix_write ERROR");
  close(sh[0]);
  return -4;
 }

pthread_t response_listener;
 pthread_create( &response_listener, 0, (void* (*)(void *))zabbix_listen, (void *)sh );
 pthread_detach( response_listener );
```

```
return 0;
}
```

ZABBIX.H

```
#ifndef ZABBIX_H
#define ZABBIX_H

#include <sys/socket.h>
#include <netinet/ip.h>

typedef struct zabbix_msg_st zabbix_msg_t;
struct zabbix_msg_st {
  struct sockaddr_in *zabbix_server;
  int max_size;

  char *data;
  int size;

  char is_item;
};

void zabbix_message_reinit(zabbix_msg_t *msg);

void zabbix_message_add_item(zabbix_msg_t *msg, const char *host, const char *item, const ⤸
char *value);
void zabbix_message_add_item_t(zabbix_msg_t *msg, const char *host, const char *item, const ⤸
char *value, int time);
void zabbix_message_add_item_f(zabbix_msg_t *msg, const char *host, const char *item, float ⤸
value);
void zabbix_message_add_item_tf(zabbix_msg_t *msg, const char *host, const char *item, float ⤸
value, int time);

int zabbix_message_finish(zabbix_msg_t *msg);
int zabbix_write(int fd, zabbix_msg_t *msg);

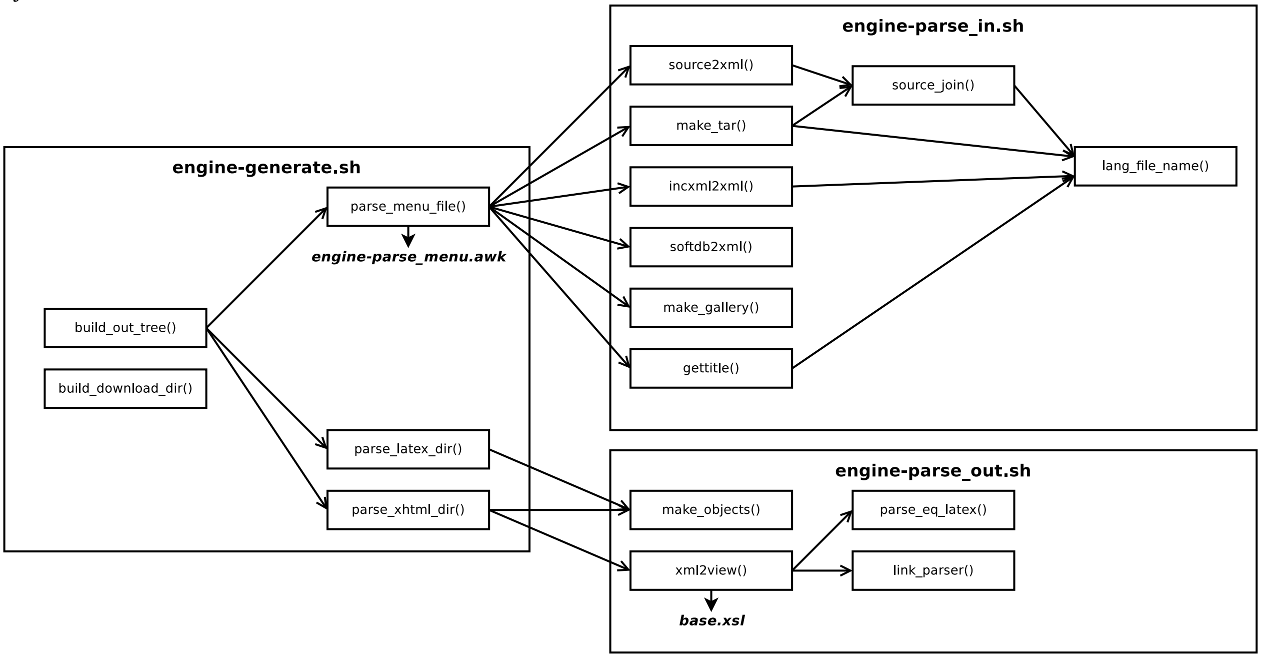
int zabbix_listen(int *sh);
int zabbix_send(zabbix_msg_t *msg);

#endif
```

Silnik WWW

Głównym elementem systemu obsługującego stronę jest zestaw plików składających się na rozbudowany skrypt bashowo-awkowy. Podstawowymi jego elementami są: engine-base.sh, engine-generate.sh, engine-parse_menu.awk, engine-parse_in.sh, engine-parse_out.sh, engine-config.sh. Funkcje zawarte w tych plikach odpowiedzialne są za budowanie wynikowego drzewa katalogowego strony wraz z konwersją i generacja plików wchodzących w skład strony. Na poniższym schemacie przedstawione są zależności (przebieg wywołań) najważniejszych z tych funk-

cji:



Dodatkowo w skład tego zestawu wchodzi pliki z funkcjami pomocniczymi: engine-edit_help.sh. Wspomagającą wobec tego zestawu skryptów funkcje pełni skrypt PHP (system.php) odpowiedzialny za konwersję niektórych z elementów strony w locie (w trakcie realizacji odpowiedzi na zapytanie HTTP przez serwer WWW) wraz z plikiem konfiguracyjnym serwera WWW ("httpd.conf") odpowiedzialnym m.in. za przekierowania po zmianach organizacji strony.

Plik silnika jest skryptem bashowym stanowiącym zbiór funkcji odpowiedzialnych za generowanie nagłówków, stopek, informacji o prawach autorskich, konwersje dokumentów, budowanie drzewa katalogowego strony wraz z zawartością oraz archiwizację. Działanie poszczególnych funkcji oraz ich parametry opisane są w komentarzu poprzedzającym deklarację tej funkcji. Istotne elementy funkcji oparte są na wywołaniach sed'a oraz awk'a (m.in. analiza pliku "menu.xml" oraz wywoływanie odpowiednich funkcji generujących). Argumenty skryptu analizowane są w oparciu o skrypt awk i basho'wą konstrukcję `switch`. Skrypt umożliwia wywołanie dowolnej z funkcji składającej się na niego (parametr -f) oraz wywołanie zbiorów funkcji odpowiedzialnych za generowanie, archiwizację czy też upload strony (np. parametr -b). Zbiory takie wraz z parametrami je wywołującymi deklarowane są w pliku konfiguracyjnym w funkcji `run_conf`.

Silnik umożliwia także budowanie wersji pdf. Konwersją XML'a na XHTML lub LaTeX oraz obsługą bazy oprogramowania zajmują się arkusze XSL - base.xsl, softdb.xsl.

Wspomagający plik system.php (na serwerze umieszczany jako system.system) odpowiedzialny jest za:

— obsługę wersji nie korzystającej z vhostów (głównie modyfikacja linków globalnych)
— generowanie wersji w kodowaniu ASCII (opcja URL: `?no_pl` lub `?no_pl&page=/page/dir`)
— obsługę linkowania po ID z przekierowaniem na odpowiednią zakładkę (opcja URL: `?id_link=page:id`)
— obsługę linkowania po ID bez przekierowania (opcja URL: `?no_redirect&id_link=page:id`)

Głównymi plikami konfiguracyjnymi silnika są: "menu.xml" oraz "OpCode.conf". Dodatkowo niektóre ustawienia umieszczone są także bezpośrednio w "engine.sh" oraz "silnik.php" i "httpd.conf". Plik "OpCode.conf" zawiera zmienne konfiguracyjne silnika strony i musi on być podawany jako pierwszy parametr wywołania skryptu (określa także położenie pliku "menu.xml"). Plik "menu.xml" jest plikiem XML'owym opisującym strukturę generowanej strony (przy pomocy różnych plików menu.xml z tych samych źródeł - w tym celu wywołujemy silnik z alternatywnym plikiem konfiguracyjnym lub tylko plikiem menu '/engine.sh -f build_out_tree "pelna/sciezka/do/alternatywnego/menu.xml"'). Składnia pliku "menu.xml" (w [] podano argumenty opcjonalne):

— `<dir name="nazwa_katalogu_out" [title ="Tytuł strony"] [path="sciezka do dokumentu"] [inmenu="0"] [inmap="0"] [menu="..."] [pdf_base_level="x"] ...> ... </dir>`
  — wszystkie pozostałe znacznik muszą być wewnątrz jakiś `<dir>...</dir>`
  — znacznik powoduje utworzenie katalogu o podanej nazwie w katalogu określonym nadrzędnym `<dir />`
  — znacznik (jeżeli nie podano `inmenu="0"` / `inmap="0"`) powoduje wstawienie pozycji w menu / mapie strony
  — atrybut `title` okresla tutuł strony który będzie użyty w menu i mapie strony
  — jeżeli podano `path` to tytuł zostanie pobrany z pierwszego znacznika `<title>` wskazanego dokumentu, ponaddto system zachowa się tak jakby wystapil znacznik `<file>` z tym dokumentem (mozliwe jest podanie atrybutów charakterystycznych dla tego znacznika)

- atrybut `menu` umozliwia wyłączenie wyświetlania menu na stronie danego katalogu (wartość ”no”), tego katalogu i wszystkich jego podkatalogów (wartość ”never”), a także wskaznie menu z innego katalogu które ma zostać umieszczone (ścieżka względna względem bierzącego katalogu)
- atrybut `pdf_base_level` określa poziom głównego tytułu (w xhtml <h1>) danego artykułu w PDFie, może przyjjmować wartości bezwzględne lub postaci +x, domyślnie +1

- `<gallery name="nazwa_katalogu_out" path="siezka/do/katalogu/zrodlowego" /> -` utworzenie wskazanego katalogu, zawierającego galerii zdjęć utworzoną z podanego katalogu źródłowego
- `<file path="sciezka/do/wlaczanego/fragmentu/strony/glownej/aktualnego/katalogu" [level="x"] [title="tytuł używany w menu artykułu"] [name="id_uzywane_w_linkowaniu"/>]`, path może zawierać zmienne środowiskowe w postaci $ZMIENNA (ale nie ${ZMIENNA}), path może wskazywać ścieżkę bezwzględną, id powinno być unikalne w skali strony, gdy nie zostanie podane utworzone zostanie na podstawie `path=""`, atrybut `level` określa poziom tytułu włączanego pliku w pliku wynikowym, zaleca się nie stosowanie manualnego ustawiania id ze względu na problemy z konwencją systemu id_link
- `<soft_db name="kategoria" [level="x"] /> -` wstawienie spisu oprogramowania z podanej kategorii, atrybut `level` tak samo jak w `<file>`
- `<text>`dowolny tekst włącznie z znacznikami HTML (mogą być zmiany w spacjach przy znacznikach <> i ”) zostanie włączony do strony aktualnego katalogu</text>
- `<addtar name="katalog/tarowania" path="lista/sciezek do/tarowanych/plikow rozdzielana spacjami" />`
- `<tar [show_dir="1"] name="katalog/tarowania=nazwa_archiwum" path="lista/sciezek do/tarowanych/plikow rozdzielana spacjami" />`

  powoduje utworzenie archiwum z podanego katalogu (nazwa archiwum to nazwa_katalogu.tgz,

  tarowane są też pliki/podkatalogi dodane do tego katalogu wcześniejszymi <addtar />

  po tarowaniu katalog jest usuwany, chyba że podano show_dir=”1” (wtedy jest przenoszony do katalogu o nazwie określonej przez atrybut name z dodanym na końcu ”.notar” umieszczonego w aktualnie budowanym katalogu strony)
- `<cmd name="polecenie" />`
  powoduje wykonanie polecenia w trakcie przetwarzania pliku menu, znaki specjalne (< > ”) wchodzące w skład polecenia należy kodować jako encje (odpowiednio: < > ")
- `<menutext>jak znacznik text tyle ze tekst zostanie dodany do treści menu</menutext>`
- `<comments [type="TREND|TODO"] text="komentarz znacznikowy, nie może zawierać innych znaczników, zaleca się komentarz umieszczać jako zawartość atrybutu, nie jest to jednak wymagane" />`
- `<!-- komentarz blokowy, może zawierać znaczniki, nie może zwierać innych komentarzy tego typu -->`

Pliki konfiguracyjne danego zasobu (”body.info”) zrealizowane są w składni shellowej i mogą zawierać następujące deklaracje zmiennych:

```
inhead="${inhead}DODATKOWE INFORMACJE DO NAGLOWKA XHTML"
ext="TYP PLIKU ZRODLOWEGO"
years="LATA DO NOTY COPYRIGHT"
coauthors="LISTA DODATKOWYCH AUTORÓW"
copyright_add="${copyright_add} DODATKOWY TEKST COPYRIGHT"
copyright_mode="special" # powoduje że nota ustawiona w copyright_add pojawi się
↪  zamiast standardowej, coauthors jest wtedy ignorowane
```

Plany rozwojowe silnika:

- system słów kluczowych i jakiegoś (hierarchicznego) indeksu
- tag en=”” w znacznikach <wiki /> powodujący dodanie w formie indeksu gornego (en) linku do wikipedii en
- nowy system generacji paczek deb i udoskonaleniem systemu tar’owania:
  - w menu.xml tag <deb base=”(katalog zawierajacy podstawowy dla generowanej paczki - przedewszystkim podkatalog debian i src)” files=”(pliki kopiowane do src)”>, albo stosowne atrybuty w tagu <tar />
  - być może atrybut add_to_tar w tagu <dir /> aby ograniczyc podwojne generowanie
  - być może generowanie tar’ów w TMP i twarde linkownie w PAGE_DIR
  - przygotowywanie na etapie build_out_tree, a wykańczanie na etapie build_download_dir

ENGINE-BASE.SH

```
#!/bin/bash

debug() {
 base_filter() {
  grep --line-buffered -v "Reading input document from stdin"
}
 if [ "$VERBOSE" = "1" ]; then
  "$@" 2>&1 | base_filter | grep --line-buffered -B1 -v "^"`echo -e "\\033"`"\\[34m" | grep --⤸
  line-buffered -v "^--"
elif [ "$VERBOSE" = "0" ]; then
  "$@" 2>&1 | base_filter | grep --line-buffered -B1 -v "^"`echo -e "\\033"`"\\[3[234]m" | ⤸
  grep --line-buffered -v "^--"
else
  "$@" 2>&1 | base_filter
fi
}

usage() {
  echo "USAGE for exec script function:" >&2
  echo "    $script_name -f func_name func_args" >&2
  echo "    $script_name --function func_name func_args" >&2
  echo
  echo "You can specify multipe options, for example:" >&2
  echo "    -buc OR -b -u -c OR --build --upload --copy" >&2
  echo "In multipe options can't use -f OR --function" >&2
  echo
  echo -ne "INCOMPLETE FUNCTIONS LIST (full list and documentation in source of script):\n" >⤸
  &2
  echo "$FUNCTIONS_HELP" >&2
  echo
  echo "For verbose mode use -v (--verbose) OR -vv (--more-verbose) for more verbose" >&2
  echo "For don't remove temporary files use -V" >&2
  echo "This not work with -h or -f options" >&2
}

unload_config_and_exit() {
 # usuniecie zmiennych konfiguracyjnych
 export -n SRC_DIR PAGE_DIR TMP_DIR SCRIPT SITE_MAP_DIR OUT_MODE
exit ${RETURN_CODE-2}
}

##########################
#        WYWOLANIE SKRYPTU        #
##########################

# jezeli skrypt wywolany lub includowany z wywolanego skryptu
# a nie includowany bezposrednio z powloki interaktywnej
if [ "$0" != "/bin/bash" ]; then
 EXIT="exit"

# sciezka wywołania do skryptu
 if [ -x "$0" ]; then
  SCRIPT="\"$0\""
else
  SCRIPT="bash \"$0\""
fi

# wykonanie funkcji ustalającej zmienne konfiguracyjne
```

```
 set_variable

# eksport najważniejszych zmiennych konfiguracyjnych
 export SRC_DIR PAGE_DIR TMP_DIR SCRIPT SITE_MAP_DIR OUT_MODE

# utowrzenie katalogu tymczasowego
 mkdir -p "$TMP_DIR"

# użycie źródłowych wersji skryptów *2pdf.sh gdy nie ma w systemie
 if ! which gsch2pdf.sh > /dev/null; then
  gsch2pdf.sh() { bash $SCH2PDF "$@"; }
 fi
 if ! which pcb2pdf.sh > /dev/null; then
  pcb2pdf.sh() { bash $PCB2PDF "$@"; }
 fi
 if ! which dia2pdf.sh > /dev/null; then
  dia2pdf.sh() { bash $DIA2PDF "$@"; }
 fi

# pomoc pełna (z informacjmi z konfiguracji)
 if [ "$1" = "-h" -o "$1" = "--help" ] ; then
  check_programs
  script_name=`basename "$0"`
  usage_conf
  usage
  unload_config_and_exit
fi

# opcja wywolania pojedynczej funkcji z pliku
 if [ "$1" = "-f" -o "$1" = "--function" ] ; then
  shift
  "$@"
  unload_config_and_exit
fi

# zamaiana argumentów postaci -xyz na -x -y -z dla potrzeb dalszego parsowania
 # wypisywanie x zabezpiecza gdy opcja jest opcją echo
 opcje=`echo $@ | awk '
 BEGIN {
  RS=" "
  FS="";
 }
 $1=="-" && $2!="-" {
  for(i=2; i<=NF; i++)
   printf("-%s ", $i)
  next
 }
 $1=="-" && $2=="-" {
  printf("--")
  for(i=3; i<=NF; i++)
   printf("%s", $i)
  printf(" ")
  next
 }
 {
  print $0
 }
 '`

# utworzenie tablicy argumentów nie będących opcjami (nie zaczynających się od -)
```

```
 i=0; li=-1;
 for a in $opcje; do
  if [ "${a:0:1}" != "-" ]; then
   if [ $li != $i ]; then
    argumenty[i]="$a"
    li=$i
   else
    argumenty[i]="${argumenty[i]} $a"
   fi
  else
   i=$(( $li+1 ))
  fi
 done

# zamiana argumentów na zmienne sterujące (pozwala na podawanie argumentów w dowolnej ↩
 kolejności)
 # tutaj tylko poziom głośności itp - pozostałe w analogicznej pętli w run_conf()
 VERBOSE=0;
 REMOVE_XML="true";
 for a in $opcje; do
  if [ "$a" = "-v" ] ; then
   VERBOSE=$(( $VERBOSE + 1 ))
  elif [ "$a" = "--verbose" ] ; then
   VERBOSE=1;
  elif [ "$a" = "--more-verbose" ] ; then
   VERBOSE=2;
  fi
  if [ "$a" = "-V" ] ; then
   REMOVE_XML="false";
  fi
 done


# przejscie do katalogu z zrodlami
 cd "$SRC_DIR"

# kontrola
 if ! check_programs; then unload_config_and_exit; fi
 #if ! check_dir_names; then unload_config_and_exit; fi
 #if ! check_file_names; then unload_config_and_exit; fi
 #if ! check_file_names2; then unload_config_and_exit; fi
 #if ! check_section_names; then unload_config_and_exit; fi

# wywołanie funkcji działania zdefiniowanej w pliku konfiguracyjnym
 run_conf


# zakonczenie
 unload_config_and_exit
else
 EXIT=""
fi
```

ENGINE-GENERATE.SH

```
#############################
#        FUNKCJE - GENERACJA        #
#############################
```

```
FUNCTIONS_HELP="$FUNCTIONS_HELP"

# $1 - kompleksowy plik menu
parse_menu_file() {
 mkdir -p "$PAGE_DIR/$SITE_MAP_DIR"
echo -n "<head1>Mapa serwisu OpCode</head1><ul>" > "$PAGE_DIR/$SITE_MAP_DIR/body.xml"
:> "$PAGE_DIR/id_index"

sed -f <(echo '
  # obsluga ewentualnych pustych dir
  s#<dir\([^>]*\)/>#<dir\1></dir>#g

  # odpowiednie ułożenie spacji przy argumentach podawanych do atrybutów znacznika
  s#"# " #g
  s#[ \n\t]*=[ \n\t]*"# =" #g

  # spacja po znaczniku otwierającym komentarz oraz po i przed zamykającym
  s#<!--#<!-- #g
  s#-->#  --> #g

  # poprzedzenie zamykającego /> lub > spacją
  s#\(/*\) *>#  \1>#g') "$1" | gawk -f $PARSE_MENU_AWK

echo "</ul>" >> "$PAGE_DIR/$SITE_MAP_DIR/body.xml"

# usunięcie pustych wpisów z mapy
 while grep '<ul></ul>' "$PAGE_DIR/$SITE_MAP_DIR/body.xml" > /dev/null ; do
  sed -f <(echo '
   s#<ul></ul>##g
   s#<li></li>##g
  ') "$PAGE_DIR/$SITE_MAP_DIR/body.xml" > "$PAGE_DIR/$SITE_MAP_DIR/body2.xhtml"
  mv "$PAGE_DIR/$SITE_MAP_DIR/body2.xhtml" "$PAGE_DIR/$SITE_MAP_DIR/body.xml"
done
}




# funkcja przetwarzająca zadany przez $1 katalog powinien być określony ścieżką
# względem $PAGE_DIR rozpoczynającą się od katalogu podrzędnego (nie od .)
parse_xhtml_dir() {
 echo -e "\\033[34mParse $1\\033[0m"

# informacje dodatkowe włączonych dokumentów <file /> oraz przetwarzanego katalogu na ⤸
 podstawie menu.xml
 use_menu=""; inhead=""; coauthors=""; copyright_add=""; years=""; multi_body=""
title=`basename $1`
if [ -f "$1/body.info" ] ; then
  source "$1/body.info"
  coauthors=`grep "coauthors" "$1/body.info" | cut -s -f2 -d"=" | sort | uniq`
fi
 [ "$years" = "" ] && years="$STD_YEAR";

# zapamiętanie danych dla nagłówka
 header_title="$title"
header_inhead="$inhead"
header_use_menu="$use_menu"
if [ "$header_use_menu" != "STATIC" -a "$header_use_menu" != "DYNAMIC" ]; then
  header_use_menu="$DEFAULT_MENU_MODE"
```

```
fi
 header_art_menu="$art_menu"

# zapamietanie danych dla stopki
 footer_years="$STD_YEAR"
footer_coauthors="$coauthors"
footer_copyright_add="$copyright_add"

# obsługa daty modyfikacji artykułu
 footer_max_date=`grep "^time=" "$1/body.info" 2>/dev/null | cut -f2 -d= | sort -r | grep UTC ⤸
 | sed -n -e '1p'`
footer_max_date=${footer_max_date:-"?"}

# znajdowanie pliku menu
 if [ "$use_menu" != "NO" -a "$use_menu" != "NEVER" -a $OUT_MODE = "xhtml" ]; then
  # gdy chcemy menu to sprawdzamy czy
  if [ "$use_menu" = "SPECIAL" ]; then
   # chcemy specjalnego
   menu="$1/$use_menu_spec"
  elif [ -f "$1/menu.xhtml" ]; then
   # mamy standardowe w bierzącym katalogu
   menu="$1/menu.xhtml"
  else
   # mamy standardowe w jakimś nadrzednym katalogu i nie ma tam use_menu ustawionego na ⤸
   NEVER*
   for par_dir in ".." "../.." "../../.."; do
    if test -f "$1/$par_dir/body.info" && egrep 'use_menu="?NEVER' "$1/$par_dir/body.info" > ⤸
    /dev/null; then
     menu="no"
     break;
    elif [ -f "$1/$par_dir/menu.xhtml" ]; then
     menu="$1/$par_dir/menu.xhtml"
     break;
    else
     menu="no"
    fi
   done
  fi
 else
  menu="no"
fi

# ustalenie listy plików body.xml do przetworzenia
 if [ "$multi_body" != "" ]; then
  body_files="$multi_body"
elif [ -f "$1/body.xml" -o "$menu" != "no" ]; then
  body_files="body.xml"
fi

# tworzenie nagłówka menu
 if [ "$menu" != "no" ] ; then
  up_dir="$1"
  menu_head="$title:</p>"
  while [ "$up_dir" != "/" -a "$up_dir" != "." -a "$up_dir" != "" ] ; do
   up_dir=`dirname "$up_dir"`
   if [ -f "$up_dir/body.info" ] ; then
    . "$up_dir/body.info"
    if [ $up_dir != "" ]; then
     menu_head="<a class=\"int\" href=\"[[!URL""PREFIX!]]/$up_dir/[[!URLPOSTFIX!]]\">$title<⤸
     /a> -&gt; $menu_head"
```

```
      else
       menu_head="<a class=\"int\" href=\"[[!URL""PREFIX!]]/[[!URLPOSTFIX!]]\">$title</a> -&gt; ⤸
       $menu_head"
      fi
    fi
  done
  menu_head="<p class=\"menu_path\">$menu_head"

  sed -e 's#</li><lisub>#<ul class="submenu_link">\n\t<li>#g' \
      -e 's#</lisub><li>#</li>\n</ul></li>\n<li>#g' \
      -e 's#</lisub><lisub>#</li>\n\t<li>#g' \
      -e 's#</lisub><lisubsub>#<ul class="subsubmenu_link">\n\t\t<li>#g' \
      -e 's#</lisubsub><lisub>#</li>\n\t</ul></li>\n\t<li>#g' \
      -e 's#</lisubsub><lisubsub>#</li>\n\t\t<li>#g' -i $menu
 else
  menu_head="-"
fi

# przygotowywanie menu artykułu
 if [ "${header_art_menu:-0}" -ge "1" ]; then
  # gdy istnieje plik z menu artykułu używamy go (wprowadzając pewne modyfikacje)
  if [ -f "$1/art_menu.xhtml" ]; then
   sed -i -e '1i<p class="art_menu">' -e 's#</a> <a #</a> :: <a #g' "$1/art_menu.xhtml"
   echo '</p>' >> "$1/art_menu.xhtml"
  # w przeciwnym razie generujemy go z pliku z treścią artykułu
  elif [ -f "$1/body.xml" ]; then
   echo -n '<p class="art_menu">' >> "$1/art_menu.xhtml"

   tr '\n' ' ' < "$1/body.xml" | sed 's#<head[0-9] id="\([^<>"]*\)">\([^<>]*\)</\(head[0-9]\)>⤸
   #\n<art_menu>\1<art_menu>\2<art_menu>\3<art_menu>\n#g' | awk '
    BEGIN {FS="<art_menu>"; separator="";}
    $2=="" {next}
    $4=="head1" {
     printf("%s<a href=\"#%s\" class=\"art_menu1\">%s</a>", separator, $2, $3);
     separator=" :: ";
    }
    $4=="head2" {
     printf("%s<a href=\"#%s\" class=\"art_menu2\">%s</a>", separator, $2, $3);
     separator=" :: ";
    }
    $4=="head3" {
     printf("%s<a href=\"#%s\" class=\"art_menu3\">%s</a>", separator, $2, $3);
     separator=" :: ";
    }
   ' >>  "$1/art_menu.xhtml"

   echo '</p>' >> "$1/art_menu.xhtml"
  fi
 # gdy wyłączone czyścimy ewentualne śmieci w art_menu.xhtml
 else
  :> "$1/art_menu.xhtml"
fi

# gdy są standardowe dokumenty do przetworzenia to dla każdego z nich
 [ "$body_files" != "" ] && for body_file in $body_files; do
  if [ ! -e "$1/$body_file" -a "$menu" = "no" ]; then
   continue
  fi

  # ustalamy nazwę pliku wyjściowego
```

```
out_file=`expr $body_file : 'body[.]\(.*[.]\)xml'`
out_file="${out_file}page"

# wstawiamy naglowek dokumentu
if [ "$header_use_menu" = "STATIC" ]; then
 std_xhtml_header "$header_title" "[[!URL""PREFIX!]]" "static" "$header_inhead" > ⤸
 "$TMP_DIR/page.base.xhtml"
else
 std_xhtml_header "$header_title" "[[!URL""PREFIX!]]" "dynamic" "$header_inhead" > ⤸
 "$TMP_DIR/page.base.xhtml"
fi

# wstawiamy menu artykulu
[ -f "$1/art_menu.xhtml" ] && cat "$1/art_menu.xhtml" >> "$TMP_DIR/page.base.xhtml"

# wstawiamy artykul
if [ -f "$1/$body_file" ] ; then
 pushd "$1" > /dev/null
 xml2view "$body_file" >> "$TMP_DIR/page.base.xhtml" || mv -n "$body_file" "$body_file.⤸
 error.bck"
 popd > /dev/null
fi

# wstawiamy stopke wraz z menu
std_xhtml_fotter "$1" "$footer_max_date" "$menu" "$menu_head" "[[!URL""PREFIX!]]" \
 "$footer_years" "$footer_coauthors" "$footer_copyright_add" >> "$TMP_DIR/page.base.xhtml"

#
# w tym momęcie mamy poprawny XHTML ale zawierający [[!URL"PREFIX!]] i [[!URL"POSTFIX!]]
# które będą służyły do generacji wariantów strony
#
# znaki " (lub "") w powyzszych napisach zostały wprowadzone w treści skryptu aby napisy ⤸
te
# nie były podmieniane w trakcie generacji stron prezentujących kod silnika
#

# podstawowy wariant (linkowanie absolutne do katalogu)
sed \
 -e '# wstawienie prefiku URL zamiast stałej napisowej URLPREFIX
  s#\[\[!URLPREFIX!\]\]#'"$BASE_URL_PREFIX"'#g' \
 -e '# wstawienie nazwy pliku xhtml zamiast stałej napisowej URLPOSTFIX dla katalogów ⤸
 mających wiele plików NUM.page.xhtml
  s#[.]\[\[!URLPOSTFIX!\]\]#'.page.xhtml'#g' \
 -e '# wstawienie nazwy pliku xhtml zamiast stałej napisowej URLPOSTFIX dla pozostałych ⤸
 linków
  s#\[\[!URLPOSTFIX!\]\]#'"$LINK_PAGE_FILE"'#g' "$TMP_DIR/page.base.xhtml" > "$1/$out_file.⤸
  xhtml"

# inne warianty - linkowana względnie
URL_PREFIX=`revpath $1`
if [ "$URL_PREFIX" = "" ]; then
 URL_PREFIX="."
fi
URL_PREFIX="$BASE_URL_PREFIX$URL_PREFIX"
sed -i -e '# dodanie blokady robotów
  s#<title>#<meta name="robots" content="noindex, nofollow" /><title>#g' \
 -e '# wstawienie prefiku URL zamiast stałej napisowej URLPREFIX
  s#\[\[!URLPREFIX!\]\]#'"$URL_PREFIX"'#g' "$TMP_DIR/page.base.xhtml"

# wersja offline
```

```
  sed \
   -e '# wstawienie nazwy pliku xhtml zamiast stałej napisowej URLPOSTFIX
    s#\[\[!URLPOSTFIX!\]\]#page.OFFLINE.xhtml#g' \
    "$TMP_DIR/page.base.xhtml" > "$1/$out_file.OFFLINE.xhtml"


  # ascii
  konwert UTF8-ascii "$TMP_DIR/page.base.xhtml" |
   sed  \
   -e '# wstawienie nazwy pliku xhtml zamiast stałej napisowej URLPOSTFIX
    s#\[\[!URLPOSTFIX!\]\]#page.ASCII.xhtml#g' \
   -e '# podmiana kodowania w nagłówku <meta />
    s#<meta \([^<>]*\) charset=utf-8" />#<meta \1 charset=ascii" />#g' > "$1/$out_file.ASCII.⤸
    xhtml"

  # z statycznym menu
  #sed \
  # -e '# wstawienie nazwy pliku xhtml zamiast stałej napisowej URLPOSTFIX
  #   s#\[\[!URLPOSTFIX!\]\]#page.STATIC-MENU.xhtml#g' \
   # -e '# podmiana css na statyczne menu
  #   s#\(<link[^>]*layout\)_dynamic[.]\(css[^>]*>\)#\1_static.\2#g' \
  #   "$TMP_DIR/page.base.xhtml" > "$1/$out_file.STATIC-MENU.xhtml"
 done
 unset out_file body_file body_files menu menu_head header_art_menu\
  header_title title header_inhead inhead header_use_menu use_menu \
  footer_years footer_coauthors coauthors footer_copyright_add copyright_add

# gdy jest README.xhtml katalogu notar
 if [ -f "$1/README.notar.xhtml" ]; then
  URL_PREFIX="$BASE_URL_PREFIX$notar_url_prefix"

  sed 's#<a href="\([^"]*\)">\([^<>]*\)</a>#<a href="'"$URL_PREFIX"'/\1">'"$URL_PREFIX"'/\2<⤸
  /a>#g' \
   "$1/README.notar.xhtml" > "$1/README.xhtml"
  rm "$1/README.notar.xhtml"
fi

# generowanie obiektów tworzonych dynamicznie ze źródeł
 make_objects $1
}



# funkcja buduje plik latex z podanego w $1 katalogu
parse_latex_dir() {
 pushd "$1" > /dev/null

# uzyskujemy informacje dla stopki z body.info
 coauthors=""
copyright_add=""
# informacje dodatkowe włączonych dokumentów <file /> oraz przetwarzanego katalogu na ⤸
 podstawie menu.xml
 if [ -f "./body.info" ] ; then
  coauthors=`grep "coauthors" "./body.info" | cut -s -f2 -d"=" | sort | uniq`
fi
 years="$STD_YEAR"

# generujemy plik page.tex
 std_latex_header > "page.tex"
xml2view "body.xml" | sed -e 's#&gt;#>#g' -e 's#&lt;#<#g' -e 's#&amp;#\&#g' \
```

```
  -e '# wstawienie prefiku URL zamiast stałej napisowej URLPREFIX
   s#\[\[!URLPREFIX!\]\]#'"$URL_PREFIX"'#g' \
  -e '# wstawienie nazwy pliku xhtml zamiast stałej napisowej URLPOSTFIX
   s#\[\[!URLPOSTFIX!\]\]#'"$LINK_PAGE_FILE"'#g' >> "page.tex"
std_latex_fotter "$years" "$coauthors" "$copyright_add" >> "page.tex"

# generujemy obiekty tworzone dynamicznie ze źródeł
 make_objects .

popd > /dev/null
}



# funkcja budująca drzewo wynikowe lub generująca wersję latex i pdf
#   (w zależności od zmiennej OUT_MODE)
# $1 == menu.xml
build_out_tree() {
 # funkcja wspomagająca pdflatex w tworzeniu zakładek pdf'owych
 # jako $1 przyjmuje wychodzący z pdflatex plik .out
 correct_hierarhy_in_pdf_bookmark() {
  mv $1 $1.old
  awk '
  BEGIN {
   FS="[{}]"
   types["chapter"]=0
   types["section"]=1
   types["subsection"]=2
   types["subsubsection"]=3
  }
  # ustalamy typ rekordu bierzącego i typ podawanego mu rodzica
  {
   rtype=gensub("[*.].*", "", "g", $2)
   ptype=gensub("[*.].*", "", "g", $6)
  }
  # jeżeli bierzący rekord obniżył się gwałtownie w hierarchi to zapamiętujemy jego ⤸
  rodzica
  types[ptype] < types[rtype]-1 {parent=$6}

  # jeżeli rodzić rakordu jest tego samego typu co rekord to rodzica zastepujemy ⤸
  zapamiętanym
  rtype == ptype {
   printf("%s{%s}{%s}{%s}\n", $1, $2, $4, parent)
   next
  }
  # w przeciwnym razie przepisujemy bez zmian
  {
   print $0
  }
  ' < $1.old > $1
 }

mkdir -p "$PAGE_DIR"
mkdir -p "$CACHE_DIR"

cd "$TMP_DIR"
:> "$TMP_DIR/todo_links.xml"
case $OUT_MODE in
  "xhtml"*)
   # wygenerowanie styli highlight
```

```
echo "" | highlight --out-format=xhtml -S c > /dev/null
mv highlight.css "$PAGE_DIR"


# zbudowanie struktury strony body.xml zgodnie z menu.xml
cd "$SRC_DIR"
parse_menu_file "$1"

# przetwarzanie body.xml na page.xhtml w każdym z katalogów strony
cd "$PAGE_DIR"
find * -type d | grep -v "^~" |
while read katalog; do
 parse_xhtml_dir "$katalog"
done
parse_xhtml_dir .

# sprzątanie ...
if $REMOVE_XML; then
 echo -e "\\033[34mCleaning up output directory\\033[0m"
 find "$PAGE_DIR" -name "page.base.xhtml" -o -name "*menu.xhtml" -o -name "body.info" -o -⤸
 name "body*.xml" |
  while read f ; do rm -f "$f" ; done
else
 echo "WARNING: -V is set - we DON'T remove: page.base.xhtml body*.xml, body.info, menu.⤸
 xhtml, and art_menu.xhtml from out tree"
fi

# utworzenie dolinkowania do ~DOWNLOAD
ln -s "$DOWNLOAD_DIR" "~DOWNLOAD"
;;
"latex"*)
# wygenerowanie styli highlight
echo "" | highlight --out-format=latex -S c > /dev/null
grep -e '\definecolor' -e '\newcommand' highlight.sty > "$PAGE_DIR/highlight.sty"

# przygotowanie wersji menu.xml dla pdf'owanej wersji strony
# remove <* pdf="no>
# replace <pdf-text> by <text>
grep -v -e 'pdf="no"' -e '<tar' -e '<addtar' "$1" |
 sed -e 's#pdf-text>#text>#g' > "$TMP_DIR/OpCode_menu-PDF.xml"


# zbudowanie zbiorczego body.xml zgodnie z zmodyfikowanym menu.xml
cd "$SRC_DIR"
parse_menu_file "$TMP_DIR/OpCode_menu-PDF.xml"

# zbudowanie zbiorczego page.tex i plików pomocniczych
cd "$PAGE_DIR"
URL_PREFIX="$PAGE_URL$BASE_URL_PREFIX"
parse_latex_dir "."

# wykonywanie pdflatex
echo -e "\\033[34mGenerate pdf file\\033[0m"
if  [ "$OUT_MODE" != "latex-nopdf" ]; then
 echo
 echo "In case LaTeX error install Debian packages:"
 echo "       texlive-luatex texlive-xetex texlive-latex-recommended texlive-generic-⤸
 recommended"
 echo "       fonts-lmodern texlive-fonts-recommended texlive-lang-polish fonts-⤸
 linuxlibertine"
```

```
    lualatex page.tex
    echo
    echo "Powtorna generacja pdf - w celu uzyskania spisu tresci:"
    echo "  lualatex -interaction scrollmode page.tex"
    if  [ "$OUT_MODE" != "latex-onepdf" ]; then
     sleep 0.1
     correct_hierarhy_in_pdf_bookmark page.out
     lualatex -interaction scrollmode page.tex > /dev/null
     correct_hierarhy_in_pdf_bookmark page.out
     lualatex -interaction scrollmode page.tex > /dev/null
    fi
   fi
   ;;
 esac
}

# generacja katalogu ~DOWNLOAD, w tym:
#    * archiwum tar.gz z wersjami źródłowymi strony
#    * paczek *.deb
build_download_dir() {
 # tworzenie strony ~DOWNLOAD
 rm -fr "$DOWNLOAD_DIR"
mkdir -p "$DOWNLOAD_DIR"

if [ "$DOWNLOAD_README_1" != "" ]; then
  :> "$DOWNLOAD_DIR/README.xhtml"
  echo "$DOWNLOAD_README_1<ul>" >> "$DOWNLOAD_DIR/README.xhtml"

  for p in $NO_TAR_LINK; do
   echo "<li><a href=\"$URL_PREFIX$p\">$p</a></li>" >> "$DOWNLOAD_DIR/README.xhtml"
  done

  echo "</ul>$DOWNLOAD_README_2" >> "$DOWNLOAD_DIR/README.xhtml"
fi

if [ "$SOURCE_ARCHIVE_NAME" != "" ]; then
  # tworzenie archiwum tar z źródłami strony
  cd "$SRC_DIR"
  for p in $NO_TAR; do
   NEW_NO_TAR="$NEW_NO_TAR --exclude=$p"
  done
  tar -czh --exclude="*/.git" --exclude="*/ARCHIVE/*" --exclude="*/PHOTO/*" $NEW_NO_TAR -f ⤸
  "$DOWNLOAD_DIR/$SOURCE_ARCHIVE_NAME" .
fi

if [ "$OFFLINE_ARCHIVE_NAME" != "" ]; then
  # tworzenie archiwum tar z wersją off-line strony
  cd "$PAGE_DIR"
  tar -czh --exclude="*/.git" --exclude="*$PHOTO_DIR*" --exclude="*/archiwum/*" --⤸
  exclude="*~DOWNLOAD*" --exclude="*MIRRORS*" \
   -f "$DOWNLOAD_DIR/$OFFLINE_ARCHIVE_NAME" .
fi

cd "$TMP_DIR"

# budowanie PDFów
 echo "$MAKE_PDF" | while read path num; do
  path=`expr "$path" : '[ \t]*\(.*\)'`

  if [ -f "$PAGE_DIR/$path" ]; then
```

```
    basename=`basename "$PAGE_DIR/$path"`
    basename=${basename%.tex}

    for f in `seq 1 ${num:-1}`; do
     pdflatex "$PAGE_DIR/$path"
    done

    mv $basename.pdf "$DOWNLOAD_DIR"
  fi
 done

# budowanie DEBów
 for p in $MAKE_DEB; do
  basename=`basename "$PAGE_DIR/$p"`
  basename=${basename%.*}
  version="1.0.`date +"%Y%m%dT%H%M%S"`"

  tar -xzf "$PAGE_DIR/$p"

  pushd $basename > /dev/null
  debchange -v $version "auto build"
  dpkg-buildpackage -rfakeroot
  popd > /dev/null

  mv ${basename}_${version}_*.deb "$DOWNLOAD_DIR/${basename}.deb"
  rm -fr $basename ${basename}_${version}*
 done
}
```

ENGINE-PARSE_MENU.AWK

```
# funkcja zwracajaca sciezke do pliku (czesc nazwy przed ostatnim /)
function dirname(path_in) {
 # aby nie wyjść za wysoko
 if (path_in=="/" || path_in=="")
  return "/"

# rozkladamy sciezke
 tmp = split(path_in, tmp2, "/");

# pomijamy koncowe puste pola
 while (tmp2[tmp] == "" && tmp > 0) tmp--;

# generujemy nowa sciezke
 path_in="";
 for (i=1; i<tmp; i++)
  if (path_in == "" || path_in == "/")
   path_in=sprintf("/%s", tmp2[i]);
  else
   path_in=sprintf("%s/%s", path_in, tmp2[i]);

return path_in
}

# funkcja zwracjajaca rozszerzenie (czesc nazwy po ostatniej kropce)
function ext(path_in) {
 # rozkladamy sciezke
```

```
 tmp = split(path_in, tmp2, ".");

# pomijamy koncowe puste pola
 while (tmp2[tmp] == "" && tmp > 0) tmp--;

# zwracamy wynik gdy nie bylo kropki ciag pusty
 if (tmp <= 1)
   return ""
else
   return tmp2[tmp]
}

# funkcja zwracajaca nazwe pliku (czesc nazwy po ostatnim /)
function basename(path_in) {
 # rozkladamy sciezke
 tmp = split(path_in, tmp2, "/");

# pomijamy koncowe puste pola
 while (tmp2[tmp] == "" && tmp > 0) tmp--;

return tmp2[tmp]
}

function is_file(path_in) {
 return system( "ls >/dev/null 2>&1 " path_in ) == 0
}

BEGIN {
 RS="<"
# zamknięcia znaczników i to co po nich wyrzucamy jako nieinteresujące nas atrybuty a ⤸
 nie przez:
 # RS="(>?[^><]*<)|(>)"
 FS="[ \n\t]+"

# ustawienie wartosci poczatkowych zmiennych,
 # w tym zmiennych odpowiadających wybranym zmiennym środowiskowym
 is_comments=0; is_text=0; is_menutext=0; cur_dir="";
 SRC_DIR=ENVIRON["SRC_DIR"];
 PAGE_DIR=ENVIRON["PAGE_DIR"];
 SITE_MAP_FILE=PAGE_DIR "/" ENVIRON["SITE_MAP_DIR"] "/body.xml";
 TMP_DIR=ENVIRON["TMP_DIR"];
 SCRIPT=ENVIRON["SCRIPT"];
 OUT_MODE=ENVIRON["OUT_MODE"];

# zmienna zawierjaca komende ustawiajaca srodowisko w niektorych wywolaniach system():
 set_env=sprintf("SRC_DIR=\"%s\"; PAGE_DIR=\"%s\"; TMP_DIR=\"%s\"; SCRIPT=\"%s\"; ", SRC_DIR, ⤸
 PAGE_DIR, TMP_DIR, SCRIPT);

pdf_level_index=0
 pdf_level[0]=0
}

END {
 close(PAGE_DIR "/id_index")
 close(SITE_MAP_FILE)
}

$0=="" || $0=="\n" {
 next
}
```

```
############################
###   ZNACZNIKI "BLOKOWE"   ###
############################

# komentarze
$1=="!--" {
 is_comments = 1
}
is_comments==1 {
 for (i=1; i<NF; i++) {
  if ($i=="--" && $(++i)==">") {
   is_comments = 0
   break;
  }
 }
 next; # zapobiega realizacji kolejnych instrukcji gdy trwa komentarz
}

# teskt
$1=="text" {
 is_text = 1
 next;
}
$1=="/text" {
 printf("\n") >> PAGE_DIR "/" cur_files_dir "/body.xml"
close(PAGE_DIR "/" cur_files_dir "/body.xml")
 is_text = 0
 next;
}
is_text==1 {
 gsub(" *\"", "\"")
 gsub(" *=\" *", "=\"")
 gsub(" >", ">")
 printf("<%s", $0) >> PAGE_DIR "/" cur_files_dir "/body.xml"
next; # zapobiega realizacji kolejnych instrukcji gdy trwa text
}

$1=="/menutext" {
 close(menutext_file)
 is_menutext = 0
 next;
}
is_menutext==1 && OUT_MODE !~ "^latex" {
 gsub(" *\"", "\"")
 gsub(" *=\" *", "=\"")
 gsub(" >", ">")
 printf("<%s", $0) >> menutext_file
next; # zapobiega realizacji kolejnych instrukcji gdy trwa text
}



############################
###         ATRYBUTY         ###
############################

# atrybuty znaczników które takowe posiadają
$1=="menutext" || $1=="cmd" || $1=="dir" || $1=="file" || $1=="tar" || $1=="addtar" || ⤸
```

```
$1=="gallery" || $1=="soft_db" {
 delete ATTR; ATTR["art_menu"] = 1;
 for (i=1; i<NF; i++) {
  if ($i == "=\"") {
   # nazwa atrybutu
   tmp = $(i-1)

   # wartosc atrybutu
   attrib = ""
   for (i++; i<=NF; i++) {
    if ($i == "\"")
     break
    if (attrib == "")
     attrib = sprintf("%s", $i);
    else
     attrib = sprintf("%s %s", attrib, $i);
   }

   # zapamietanie wartosci atrybutu
   ATTR[tmp] = attrib
  }
 }
}



############################
###   ZNACZNIKI SPECJALNE   ###
############################

# menu teskt
$1=="menutext" {
 is_menutext = 1

if ("inmenu" in ATTR && ATTR["inmenu"] != 0)
  menutext_file = PAGE_DIR "/" cur_dir "/" ATTR["inmenu"] "/menu.xhtml"
else
  menutext_file = PAGE_DIR "/" cur_dir "/menu.xhtml"

next;
}

# komenda
$1=="cmd" {
 name = ATTR["name"]
 gsub("&gt;", ">", name)
 gsub("&lt;", "<", name)
 gsub("&amp;", "&", name)
 gsub("&quot;", "\"", name)
 printf("\033[34mExec command: %s \033[0m\n", name)
 system(set_env "cd \"" PAGE_DIR "/" cur_files_dir "\"; " name);

next
}


############################
###         KATALOGI         ###
############################
```

```
# katalog
$1=="dir" {
 # name na podstawie ostatniego członu path
 if (! ("name" in ATTR)){
  ATTR["name"] = basename(ATTR["path"])
 }

# aktualizacja zmiennej zawierającej aktualne położenie
 if (cur_dir == "" || cur_dir == "/")
  cur_dir=sprintf("/%s", ATTR["name"]);
 else
  cur_dir=sprintf("%s/%s", cur_dir, ATTR["name"]);

# TODO FIXME kontrola zgodności cur_dir i/lub znacznika path

if (OUT_MODE !~ "^latex") {
  cur_files_dir = cur_dir
  INFO_FILE = PAGE_DIR "/" cur_files_dir "/body.info"

  # utworzenie katalogu
  printf("\033[32mCreate: %s\033[0m\n", cur_dir)
  system( "mkdir -p \"" PAGE_DIR "/" cur_dir "\"" )

  # obsługa informacji dodatkowych (title i menu) dla parsera katalogowego XHTML
  if ("title" in ATTR)
   printf("title=\"%s\"\n", ATTR["title"]) >> INFO_FILE
  if ("menu" in ATTR)
   if (ATTR["menu"] == "no" || ATTR["menu"] == "0")
    printf("use_menu=\"NO\"\n") >> INFO_FILE
   else if (ATTR["menu"] == "never")
    printf("use_menu=\"NEVER\"\n") >> INFO_FILE
   else if (ATTR["menu"] == "static")
    printf("use_menu=\"STATIC\"\n") >> INFO_FILE
   else if (ATTR["menu"] == "dynamic")
    printf("use_menu=\"DYNAMIC\"\n") >> INFO_FILE
   else if (ATTR["menu"] != 0)
    printf("use_menu=\"SPECIAL\"\nuse_menu_spec=\"%s\"\n", ATTR["menu"]) >> INFO_FILE
  if ("art_menu" in ATTR)
   printf("art_menu=\"%d\"\n", ATTR["art_menu"]) >> INFO_FILE
} else {
  cur_files_dir = ""

  # wartosc domyslna dla pdf_base_level
  if (! pdf_base_level in ATTR || ATTR["pdf_base_level"] == "")
   ATTR["pdf_base_level"] = "+1"

  # obsługa stosu przechowującego aktualną wartość pdf_base_level
  pdf_level_index++
  if (substr(ATTR["pdf_base_level"], 1, 1) == "+") {
   pdf_level[pdf_level_index] = pdf_level[pdf_level_index-1] + ATTR["pdf_base_level"]
  } else {
   pdf_level[pdf_level_index] = ATTR["pdf_base_level"]
  }
 }

# obsługa dir z atrybutem path
 if ("path" in ATTR) {
  ATTR["level"] = 0
  $1 = "filedir"
}
```

```
close(INFO_FILE)
}
$1=="/dir" {
 # aktualizacja zmiennej zawierającej aktualne położenie
 cur_dir = dirname(cur_dir);

if (OUT_MODE !~ "^latex") {
  cur_files_dir = cur_dir
} else {
  # obsługa stosu przechowującego aktualną wartość pdf_base_level
  delete pdf_level [pdf_level_index]
  pdf_level_index--
 }
}


#################################
###   TRESC DOKUMENTU GLOWNEGO   ###
#################################

function make_id(id, level) {
 if (substr(cur_files_dir, length(cur_files_dir), 1) != "/") {
  sep = "/"
} else {
  sep = ""
}
 printf("%s\t%s%s\n", id, cur_files_dir, sep) >> PAGE_DIR "/id_index"

if (OUT_MODE !~ "^latex" && level != 0) {
  # obsługa znaczników id i menu artykułu
  if ("title" in ATTR) {
   printf("<a class=\"art_menu%d\" href=\"#%s\">%s</a> ", level, id, ATTR["title"]) \
    >> PAGE_DIR "/" cur_dir "/art_menu.xhtml"
   close(PAGE_DIR "/" cur_dir "/art_menu.xhtml")
  }
 }
}

# obsługa poziomu menu - na razie z atrybutu, w przyszłości być może ze struktury XMLa
function get_level() {
 if ("level" in ATTR)
  return ATTR["level"] + pdf_level[pdf_level_index]
 else
  return pdf_level[pdf_level_index]
}

# obsługa zmiennych środowiskowych wramach atrybutu path
function get_path(  path, envvar) {
 path = ATTR["path"]
 if ( substr(path, 1, 1) == "$" ) {
  envvar = gensub("[^_A-Za-z0-9].*$", "", "g", substr(path, 2));
  path = sprintf("%s%s", ENVIRON[envvar], gensub("[$][_A-Za-z0-9]*", "", "g", path))
 }
 return path
}

# wstawki dokumentu głównego katalogu oraz zawartość katalogu
$1=="file" || $1=="filedir" {
 printf("\033[34mParse: %s -> %s\033[0m\n", ATTR["path"], cur_dir)
```

```
 INFO_FILE = PAGE_DIR "/" cur_files_dir "/body.info"

head_level = get_level()
 path = get_path()

# obsługa tytułu
 if ($1=="filedir" && !("title" in ATTR)) {
  if ( is_file( "\"" SRC_DIR "/" path "/body\"*.xml" ) ) {
   cmd = SCRIPT " -f gettitle " path
   cmd | getline ATTR["title"]
  } else if ( is_file( "\"" SRC_DIR "/" path "/body\"*.src" ) ) {
   ATTR["title"] = ATTR["name"]
  }
  printf("title=\"%s\"\n", ATTR["title"]) >> INFO_FILE
}

# obsługa indeksu
 if ("name" in ATTR && $1=="file") {
  id = ATTR["name"]
 } else {
  id = ATTR["path"]
  gsub("/", ":", id)
  gsub("^[.:]*", "", id)
 }
 make_id(id, head_level)

# obsługa plików
 if ( is_file( "\"" SRC_DIR "/" path "/body\"*.xml" ) ) {
  system( SCRIPT " -f incxml2xml \"" path "\" \"" cur_dir "\" " head_level " \"" id "\"")
 } else if ( is_file( "\"" SRC_DIR "/" path "/body\"*.src" ) ) {
  system( SCRIPT " -f source2xml \"" path "\" \"" cur_dir "\" " head_level " \"" id "\" - " ⤸
  ext(path) " " ATTR["download"] )
 } else {
  printf("\033[33mWARNING: no find body file in %s !!!\033[0m\n", path) > "/dev/stderr"
}

# obsługa daty modyfikacji artykułu
 system( "env TZ=UTC ls -ld --time-style=+\"|time='%Y-%m-%d %H:%M:%S (%Z)'|\" \"" SRC_DIR "/" ⤸
path "/body\".* \
  2> /dev/null | cut -f2 -d\"|\" >> \"" INFO_FILE "\"")

# obsługa informacji dodatkowych (inhead)
 if ( system( "test -f \"" SRC_DIR "/" path "/body.info\"" ) == 0 ) {
  system( "cat \"" SRC_DIR "/" path "/body.info\" >> \"" INFO_FILE "\"" )
  printf("\n") >> INFO_FILE
}

# linkowanie pozostałych plików z wskazanego katalogu
 system( "find \"" SRC_DIR "/" path "\" -maxdepth 1 -type f ! -name body*.xml ! -name body*.⤸
 src \
  ! -name TODO.tmp.txt ! -name intro*.src ! -name body.info ! -name *.xhtml ! -name body.⤸
  translate.sed \
  | while read f ; do ln -srf \"$f\" \"" PAGE_DIR "/" cur_files_dir "\" ; done" )

# linkowanie targetów linków znajdujących się w wskazanym katalogu
 system( "find \"" SRC_DIR "/" path "\" -maxdepth 1 -type l ! -name body*.xml ! -name body*.⤸
 src \
  ! -name TODO.tmp.txt ! -name intro*.src ! -name body.info ! -name *.xhtml ! -name body.⤸
  translate.sed \
  | while read l; do f=`readlink $1`; ln -srf \"" SRC_DIR "/" path "/$f\" \"" PAGE_DIR "/" ⤸
```

```
  cur_files_dir "\" ; done" )

close(INFO_FILE)
}

# spisy oprogramowania
$1=="soft_db" {
 printf("\033[34mParse soft list %s -> %s\033[0m\n", ATTR["name"], cur_dir)

id = "softdb:" ATTR["name"]
 head_level = get_level()
 make_id(id, head_level)

system( SCRIPT " -f softdb2xml \"" ATTR["name"] "\" \"" cur_dir "\" \"" head_level "\" \"" ⤸
id "\"" )

next
}

# galerie obrazków
$1=="gallery" {
 printf("\033[34mParse: %s -> %s\033[0m\n", ATTR["path"], cur_dir)

system( SCRIPT " -f make_gallery \"" get_path() "\" \"" cur_dir "\"")
 next
}



############################################
###    OPERACJE DODATKOWE TYLKO DLA XHTML    ###
############################################
OUT_MODE ~ "^latex" {
 next
}

##################################
###        ARCHIWA, PACZKI ITP        ###
##################################

# generowane archiwa *.tar.gz
function make_tar(name, show_dir, tar_path, add_path) {
 system( SCRIPT " -f make_tar \"" name "\" \"" show_dir "\" \"" tar_path "\" \"" add_path ⤸
"\"" )
}
$1=="tar" {
 printf("\033[34mParse TAR: %s -> %s\033[0m\n", ATTR["path"], ATTR["name"])
 if ("show_dir" in ATTR && ATTR["show_dir"] == 1)
  show_dir = sprintf("%s/%s.notar", cur_dir, ATTR["name"])
 else
  show_dir = "-"
make_tar(ATTR["name"], show_dir, cur_dir "/" ATTR["name"] ".tgz",  ATTR["path"])
 next
}
$1=="addtar" {
 printf("\033[34mParse TAR: %s -> %s\033[0m\n", ATTR["path"], ATTR["name"])
 make_tar(ATTR["name"], "-", "-",  ATTR["path"])
 next
}
```

```
##################################
###        MENU I MAPA SERWISU         ###
##################################

# obsługa dodawania dir i notar do menu i mapy
$1=="dir" || $1=="filedir" || ($1=="tar" && "show_dir" in ATTR && ATTR["show_dir"] == 1) {
 if ($1=="tar") {
  ATTR["name"]=sprintf("%s.notar", ATTR["name"]);
  cur_dir=sprintf("%s/%s", cur_dir, ATTR["name"]);
 }

# obsługa menu
 if (!("inmenu" in ATTR) || ATTR["inmenu"] != 0) {
  if (! "title" in ATTR || ATTR["title"] == "") {
   #printf("\033[33mWARNING: menu entry (%s) use name instead title !!!\033[0m\n", cur_dir) > ⤸
   "/dev/stderr"
   ATTR["title"] = ATTR["name"]
  }
  if (("inmenu" in ATTR) && ATTR["inmenu"] ~ /[./]*/) {
   upper = ATTR["inmenu"]
   if (ATTR["inmenu"] ~ /[.]*[/][.]*/)
    li_class = "subsub"
   else
    li_class = "sub"
  } else {
   upper = ""
   li_class = ""
  }
  printf("<li%s><a class=\"int\" href=\"[[!URL""PREFIX!]]%s/[[!URL""POSTFIX!]]\">%s</a></li%s>⤸
  ",
   li_class, cur_dir, ATTR["title"], li_class) >> PAGE_DIR "/" cur_dir "/" upper "/../menu.⤸
   xhtml"
  close(PAGE_DIR "/" cur_dir "/" upper "/../menu.xhtml")
 }

# obsługa mapy serwisu
 if (!("inmap" in ATTR) || ATTR["inmap"] != 0) {
  if (! "title" in ATTR || ATTR["title"] == "") {
   #printf("\033[33mWARNING: map entry (%s) use name instead title !!!\033[0m\n", cur_dir) > ⤸
   "/dev/stderr"
   ATTR["title"] = ATTR["name"]
  }
  printf("\n<li><a class=\"int\" href=\"[[!URL""PREFIX!]]%s/[[!URL""POSTFIX!]]\">%s</a>",
   cur_dir, ATTR["title"]) >> SITE_MAP_FILE
}

# dalszy ciąg obsługi mapy - specyficzny dla dir / tar
 if ($1=="tar") {
  if (!("inmap" in ATTR) || ATTR["inmap"] != 0) {
   printf("</li>") >> SITE_MAP_FILE
  }
  cur_dir = dirname(cur_dir);
 } else if ($1=="dir" || $1=="filedir") {
  if (!("inmap" in ATTR) || ATTR["inmap"] != 0) {
   printf("<ul>") >> SITE_MAP_FILE
  } else {
   printf("<li><ul>") >> SITE_MAP_FILE
  }
 }
```

```
next
}
# zamykanie <dir /> w mapie serwisu
$1=="/dir" {
 printf("</ul></li>") >> SITE_MAP_FILE
next
}
```

ENGINE-PARSE_IN.SH

```
########################################
#        FUNKCJE - PRZETWARZANIE PLIKÓW        #
########################################

FUNCTIONS_HELP="$FUNCTIONS_HELP"

# funkcja zwraca ścieżkę do odpowiedniej wersji językowej lub wersji uniwersalnej
# $1 - katalog z plikiem do przetwarzania
# $2 - nazwa pliku to przetworzenia $3    - rozszerzenie pliku to przetworzenia
lang_file_name() {
 if [ -e "$1/$2.$WWW_LANG.$3" ]; then
  echo "$1/$2.$WWW_LANG.$3";
 elif [ -e "$1/$2.$3" ]; then
  echo "$1/$2.$3";
 else
  echo "/dev/null"
fi
}

# $1 - katalog z plikem z którego zwrócić wartość pierwszego znacznika <title>
gettitle() {
 awk '
  BEGIN {
   RS="<title>"; FS="</title>"
  }
  NF>=2 {
   print $1
  }
 ' "$(lang_file_name "$SRC_DIR/$1" body xml)" | tr '\n' ' '
}


####################################
#        PRZETWARZANIE *.src DO XML        #
####################################

# funkcja łączy body.src, intro.src oraz noty copyright
# $1 - katalog z plikami do przetwarzania
# $2 - typ
# $3 - czy wstawiać Copyright
# $4 - podaje informacje o dacie dla Copyright
# $5 - współautorzy
# $6 - dodatkowe informacje Copyright
#
# jeżeli $3 == no to $4 $5 $6 są ignorowane
#
# jeżeli $3 == yes to $4 jest obowiązkowy
#                              $5 i $6 są opcjonalne (ale $6 implikuje podanie $5)
```

```
#
# jeżeli $3 == special to ostatni podany argument zawiera specjalną (pełną) notę ⤸
Copyright
#                                    jeżeli podane zostały jakieś argumenty pomiędzy $3 a ${!#} to ⤸
są ignorowane
source_join() {
 file_name=$(lang_file_name "$SRC_DIR/$1" body src)

# obsluga zastepowania /**TRSTRING**/ poprzez napisy z pliku tlumaczen
 # /**TRSTRING**/ powinno być umieszczane w ramach stałych napisowych "", ''
 # lub komentarzy innych niż komentzre C (/**/)
 # w przypadku komentarzy C powinno być umieszczane zamiast nich)
 # gdyż samo stanowi komentarz języka C
 : > $TMP_DIR/body.translate.sed
 if [ "`basename $file_name`" = "body.src" -a -f $SRC_DIR/$1/body.translate ]; then
  # format pliku body.translate:
  #    TR_STRING1 LANG /* komentarz C */
  #    TR_STRING2 LANG inny teskt
  while read tr lang str; do
   if [ "$lang" = "$WWW_LANG" ]; then
    echo "s#/**$tr**/#${str//#/\\#}#g" >> "$TMP_DIR/body.translate.sed"
   fi
  done < body.translate
fi

pierwsza=`sed -n -e '1p' "$file_name" | egrep '(^#!)|(<\?php)'`
if [ "$pierwsza" = "" ] ; then
  [ "$3" = "yes" ] && std_copyright "$2" "$4" "$5" "$6"
  [ "$3" = "special" ] && special_copyright "$2" "${!#}"
  cat "$(lang_file_name "$SRC_DIR/$1" intro src)"
  sed -f "$TMP_DIR/body.translate.sed" "$file_name"
else
  echo "$pierwsza"
  [ "$3" = "yes" ] && std_copyright "$2" "$4" "$5" "$6"
  [ "$3" = "special" ] && special_copyright "$2" "${!#}"
  cat "$(lang_file_name "$SRC_DIR/$1" intro src)"
  sed -e '1d' -f "$TMP_DIR/body.translate.sed" "$file_name"
fi
}

# przetworzrenie źródeł skryptu/programu określonego przez $1 na xml
#    (zawierający xhtml, latex lub inne kodowanie kodu)
# $1 - katalog z plikiem do przetwarzania $2 - katalog wynikowy $3 - poziom nagłówka $4 - ⤸
id
# $5 - extra rozszerzenie $6 - typ pliku $7 - czy generować plik do pobrania
source2xml() {
 cd "$SRC_DIR"
dir_name="$1"
ext="$6"
if [ -f "$dir_name/body.info" ]; then
  source "$dir_name/body.info"
fi

case $OUT_MODE in
 "xhtml"*)
  # przygotowanie informacji o pliku do pobrania
  src_link_name=`basename "$dir_name"`
  if [ "$5" != "-" ] ; then
   ext_ext="$5"
  else
```

```
    ext_ext=""
  fi

  # przygotowanie pliku do pobrania
  if [ "$7" = "no" ]; then
    info=""
  else
    info="<p class=\"fotter\">XHTML generated by highlight (http://www.andre-simon.de/)
    from <a href=\"./$src_link_name$ext_ext\">$src_link_name$ext_ext</a></p>"

    source_join "$dir_name" "$ext" "${copyright_mode:-"yes"}" "$years" "$coauthors" ⤸
    "$copyright_add" \
      > "$PAGE_DIR/$2/$src_link_name$ext_ext"
  fi

  # wstawienie nagłówka
  if [ "$7" != "no" ]; then
    echo "<head$3 id=\"$4\">`basename "$dir_name"`</head$3>" >> "$PAGE_DIR/$2/body.xml"
  else
    echo "<div id=\"$4\" />" >> "$PAGE_DIR/$2/body.xml"
  fi

  # wstawienie kodów xhtml w ramach <pre />
  echo "<pre>" >> "$PAGE_DIR/$2/body.xml"
  if [ "$ext" = "conf" ]; then ext="txt"; fi
  source_join "$dir_name" "$ext" "no" | sed -e 's#\t#  #g' | highlight -f --force -S $ext --⤸
  out-format=xhtml >> "$PAGE_DIR/$2/body.xml"
  echo "</pre>$info" >> "$PAGE_DIR/$2/body.xml"
  ;;
 "latex"*)
  # wstawienie nagłówka
  if [ "$7" != "no" ]; then
    echo "" >> "$PAGE_DIR/body.xml"
    echo "<head$3 id=\"$4\">`basename "$dir_name"`</head$3>" >> "$PAGE_DIR/body.xml"
  else
    echo "<div id=\"$4\" />" >> "$PAGE_DIR/body.xml"
  fi

  # wstawienie kodów latex w ramach <latex />
  echo "<latex>" >> "$PAGE_DIR/body.xml"
  [ "$ext" = "conf" ] && ext="txt"
  source_join "$dir_name" "$ext" "no" | highlight -f --force -S $ext --out-format=latex --⤸
  line-length=93 --wrap-simple -K small |
    sed -e 's#&#\&amp;#g' -e 's#<#\&lt;#g' >> "$PAGE_DIR/body.xml"
  echo -n '</latex>' >> "$PAGE_DIR/body.xml"
  ;;
 esac
}

###########################################
#       PRZETWARZANIE ZBIORU *.src DO TAR       #
###########################################

# $1 - nazwa katalogu pakowanego
# $2 sciezka do umieszczenia postaci rozpakowanej (- gdy nie zamieszczać)
# $3 - sciezka do pliku wynikowego (- gdy nie budowac)
# $4 ... $N lista plikow do dodania do katalogu
make_tar() {
 cd "$SRC_DIR"
name="$1"
```

```
tar_name=`echo $1 | cut -f1 -d'/'`
out_tar="$3"
out_dir="$2"

mkdir -p "$PAGE_DIR/$name"

shift 3
 for f in $@ ; do
  base_name=`basename "$f"`
  if [ -f "$SRC_DIR/$f/body"*.src ] ; then
   ext=`echo $base_name | awk 'BEGIN {FS="."} {print $NF}'`
   if [ -f "$SRC_DIR/$f/body.info" ]; then
    source "$SRC_DIR/$f/body.info"
#       echo "coauthors=$coauthors" >> "$PAGE_DIR/$tar_name/body.info"
#       echo "copyright_add=$copyright_add" >> "$PAGE_DIR/$tar_name/body.info"
   fi

   source_join "$f" "$ext" "yes"  "$years" "$coauthors" "$copyright_add" \
    > "$PAGE_DIR/$name/$base_name"
   file_name=`lang_file_name "$SRC_DIR/$f" body src`
   touch -m -r "$file_name" "$PAGE_DIR/$name/$base_name"
  elif [ -e "$SRC_DIR/$f" -a ! -e "$PAGE_DIR/$name/$base_name" ] ; then
   ln -sf "$SRC_DIR/$f" "$PAGE_DIR/$name/$base_name"
  fi
 done

if [ "$out_tar" != "-" ] ; then
  cd "$PAGE_DIR"
#    if [ -f "$PAGE_DIR/$name/body.info" ]; then
#      coauthors="`grep "coauthors" "$PAGE_DIR/$name/body.info" | cut -s -f2 -d"=" | sort | ⤸
uniq`"
#      rm "$PAGE_DIR/$name/body.info"
#    else
#      coauthors=""
#      copyright_add=""
#    fi
#    echo -e "$TAR_COPYRIGHT_INTRO" > "$name/LICENCE.txt"
#    std_copyright "txt" "$STD_YEAR" "$coauthors" >> "$name/LICENCE.txt"
#    touch -m -t "$TAR_TIME" "$PAGE_DIR/$name/LICENCE.txt"

  find "$name" -type d -exec touch -m -t "$TAR_TIME" \{\} \;
  tar -ch --exclude="*.svn*" --exclude="*README.notar.xhtml" -f "tmp.tar" "$name"
  gzip -n "tmp.tar"
  mv "tmp.tar.gz" "$PAGE_DIR/$out_tar"

  if [ "$out_dir" != "-" ] ; then
#      notar_copyright_intro "$out_tar" > "$name/README.notar.xhtml"
#      std_copyright "htm" "$STD_YEAR" >> "$name/README.notar.xhtml"

   mv -f "$name" "$PAGE_DIR/$out_dir"
   echo "notar_url_prefix=`revpath $out_dir`" >> "$PAGE_DIR/$out_dir/body.info"
   echo "use_menu=\"NEVER\"" >> "$PAGE_DIR/$out_dir/body.info"
  else
   rm -fr "$name"
  fi
  unset coauthors
  unset copyright_add
fi

unset name tar_name out_tar out_dir
```

```
}

###################################
#         PRZETWARZANIE *.xml DO XML        #
###################################

# sprawdza czy jest tak <TODO /> i jeżeli tak dodaje artykuł do spisu
check_todo() {
 if grep -q '<TODO' "$1"; then
  todo_title=`xmlstarlet select -t -v '//*/TODO/@title' $1`
  echo "<li><a class=\"int\" href=\"[[!URL""PREFIX!]]$2/[[!URL""POSTFIX!]]#todo\">$todo_title<⤸
  /a></li>" >> "$TMP_DIR/todo_links.xml"
fi
}

# funkcja łączy pliki xml w większą całość $1 - źródło $2 - cel $3 - poziom ⤸
nagłówka $4 - id
incxml2xml() {
 input="$(lang_file_name "$SRC_DIR/$1" body xml)"

case $OUT_MODE in
  "xhtml"*)
   check_todo "$input" "$2"
   sed \
    -e '# zamiana znacznika <title> na odpowiedni znacznik <h*>
     s%<title>\(.*\)</title>%<head'$3' id="'$4'">\1</head'$3'>%g' \
    -e '# zamiana znacznika <subtitle /> na odpowiedni znacznik <h*>
     s%<subtitle id="\([^"]*\)">\(.*\)</subtitle>%<div id="'$4':\1" /><head'$(( $3 + 1
))'>\2<⤸
     /head'$(( $3 + 1 ))'>%g' \
    -e 's%<subtitle>\(.*\)</subtitle>%<head'$(( $3 + 1 ))'>\1</head'$(( $3 + 1 ))'>%g' \
    "$input" >> "$PAGE_DIR/$2/body.xml"
   ;;
  "latex"*)
   sed \
    -e '# zamiana znacznika <title> na odpowiedni znacznik <h*>
     s%<title>\(.*\)</title>%<head'$3' id="'$4'">\1</head'$3'>%g' \
    -e '# zamiana znacznika <subtitle /> na odpowiedni znacznik <h*>
     s%<subtitle id="\([^"]*\)">\(.*\)</subtitle>%<head'$(( $3 + 1 ))' id="'$4':\1">\2<⤸
     /head'$(( $3 + 1 ))'>%g' \
    -e 's%<subtitle>\(.*\)</subtitle>%<head'$(( $3 + 1 ))'>\1</head'$(( $3 + 1 ))'>%g' \
    -e '# uzupełnienie linków do zawartości bierzącego katalogu jego pełnym położeniem ⤸
    on-line
     s%<img_from_src %<img_from_src dir="'$2'/" %g' \
    -e '# uzupełnienie linków do zawartości bierzącego katalogu jego pełnym położeniem ⤸
    on-line
     s%<sch_link %<sch_link dir="'$2'/" %g' \
    -e '# uzupełnienie linków do zawartości bierzącego katalogu jego pełnym położeniem ⤸
    on-line
     s%<pcb_link %<pcb_link dir="'$2'/" %g' \
    -e 's%href="\./\([^#]\)%href="'$2'/\1%g' \
    "$input" | grep -v '^[[:space:]]*$' >> "$PAGE_DIR/body.xml"
   ;;
 esac
}

#########################################
#         PRZETWARZANIE softdb.xml DO XML        #
#########################################
```

```
# funkcja generuje liste oprogramowania w zadanej kategorii do wyjsciowego formatu xml
# $1 - kategoria $2 - cel $3 - poziom nagłówka $4 - id
softdb2xml() {
 cd "$SRC_DIR"

case $OUT_MODE in
   "xhtml"*)
    out_dir="$2"
    ;;
   "latex"*)
    out_dir=""
    ;;
 esac

xalan -in $SOFTWARE_DB -xsl $ENGINE_DIR/softdb.xsl/body.pl.src -param cat_name "'$1'" -indent 2 | sed \
   -e '# usuniecie naglowka XML
    s#<?xml version="1.0" encoding="[UTFutf]*-8"?>##g' \
   -e '# zamiana znacznika <title> na odpowiedni znacznik <h*>
    s%<title>\(.*\)</title>%<head'$3' id=\"$4\">\1</head'$3'>%g' >> "$PAGE_DIR/$out_dir/body.xml"
}

###########################################################
#        PRZETWARZANIE KATALOGU DO GALERII MINIATUR XML        #
###########################################################

# funkcja generuje galerie obrazków w oparciu o tagi umieszczone w plikach img_name.txt
# pliki te definiuja zmienne bash'owe tags (rozdzielana spacjami lista tagów) i desc (opis zdjęcia)
# $1 - wejściowy katalog z fotkami $2 - wynikowy katalog z galeria i fotkami
make_gallery() {
 cp -r -s "$SRC_DIR/$1"/* "$PAGE_DIR/$2/"
cd "$PAGE_DIR/$2"
symlinks -rc . &>/dev/null

# zabraniamy używania menu dla podkatalogów
 echo "use_menu=\"NEVER_DOWN\"" >> ./body.info

find -name .svn -type d -exec rm -fr \{\} \; 2> /dev/null

out_file_list="body.xml"

# uzyskanie listy wszystkich plików wraz z tagami
 grep --include=*.txt -R '^tags[ \t]*=' * |
   sed -e 's%^\([^:]*\):[^=]*=\(.*\)$%\1:\2%g' -e 's%[\t ][\t ]*% %g' |
   tr -d "\"'" >  $TMP_DIR/lista_plikow.txt

# uzyskanie listy tagów
 cut -f2 -d':'  $TMP_DIR/lista_plikow.txt | tr ' ' '\n' | sort | uniq >  $TMP_DIR/lista_tagow.txt

echo "<head0>Galerie wg tagów</head0><ul>" > body.xml

# generacja galerii dla każdego z tagów
 while read tag;  do
   # nagłówek ...
   col=0
   row=0
   file_num=0
```

```
# lista plików z aktualnie przetwazranym tagiem ...
# możnaby to dac strumieniem do while read ale wtedy problem z zakresem zmiennych w pętli
grep $tag  $TMP_DIR/lista_plikow.txt | cut -f1 -d':' | sort | uniq > ⤸
$TMP_DIR/liasta_plikow_z_tagiem.txt
# dla każdego pliku z tagiem $tag ..
while read plik; do
 if [ $row -eq 0 -a $col -eq 0 ]; then
  # nagłówek ...
  out_file="body.$tag.$file_num.xml"
  echo "<head0>Galeria zdjęć dla tagu \"$tag\"</head0>" > $out_file
  echo "<table style=\"text-align: center;\">" >> $out_file
 fi
 # gdy rozpoczeliśmy nowy wiersz
 [ $col -eq 0 ] && echo "<tr>" >> $out_file

 # skalowanie obrazka
 fimg="${plik%.txt}"

 if [ -e "$fimg" -a ! -e "$fimg.small" ]; then
  suma_kontrolna=`md5sum $fimg | cut -f1 -d ' '`
  if ! grep "$fimg $suma_kontrolna" "$CACHE_DIR/check_sum.txt" > /dev/null ; then
   convert -resize 200x200 "$fimg" "$fimg.small"

   echo "$fimg $suma_kontrolna" >> "$CACHE_DIR/check_sum.txt"
   cp "$fimg.small" "$CACHE_DIR/$suma_kontrolna.small_img"
  else
   ln -sr "$CACHE_DIR/$suma_kontrolna.small_img" "$fimg.small"
  fi
 fi

 # obrazek i informacje o nim
 source "$plik"
 echo " <td>" >> $out_file
 echo "  <a href=\"$fimg\"><img src=\"$fimg.small\" alt=\"$desc\" /></a>" >> $out_file
 echo "  <br />Description: $desc" >> $out_file
 echo "  <br />Tags:" >> $out_file
 for ltag in $tags; do
  echo "   <a href=\"./$ltag.0.[[!URLPOSTFIX!]]\">$ltag</a>" >> $out_file
 done
 fparam=`TZ=UTC ls -lLh --time-style=+"%F %T UTC" "$fimg"`
 fdate=`echo $fparam | awk '{printf("%s %s %s", $6, $7, $8)}'`
 fsize=`echo $fparam | awk '{printf("%s", $5)}'`
 echo "  <br />$fsize, $fdate" >> $out_file
 echo " </td>" >> $out_file

 # zwiększenie numeru kolumny w wierszu i sprawdzenie czy nie zakończyliśmy wiersza
 col=$(( $col + 1))
 if [ $col -eq 4 ]; then
  col=0
  echo "</tr>" >> $out_file
  row=$(( $row + 1))
  if [ $row -eq 10 ]; then
   row=0
   file_num=$(( $file_num + 1))
  fi
 fi
done < $TMP_DIR/liasta_plikow_z_tagiem.txt

# zakończenie ostatniej tabelki
```

```
  if [ $col != 0 ]; then
   for col in `seq $col 3`; do
    echo '<td> </td>' >> $out_file
   done
   echo '</tr>' >> $out_file
  fi

  # stopki ...
  for out_file in body.$tag.*.xml; do
   echo "</table>" >> $out_file

   file_num=`echo ${out_file} | cut -f3 -d.`
   prev_file="$tag."$(( $file_num-1 ))
   next_file="$tag."$(( $file_num+1 ))
   echo "<p>" >> $out_file
   [ -f body.$prev_file.xml ] && echo "<a href=\"./$prev_file.[[!URLPOSTFIX!]]\">POPRZEDNIA ⤸
   STRONA</a> " >> $out_file
   [ -f body.$next_file.xml ] && echo "<a href=\"./$next_file.[[!URLPOSTFIX!]]\">NASTEPNA ⤸
   STRONA</a>" >> $out_file
   echo "</p>" >> $out_file

   out_file_list="$out_file_list $out_file"
  done;

  # obsługa listy tagów z linkami do galerii w body.xml
  echo "<li><a href=\"./$tag.0.[[!URLPOSTFIX!]]\">$tag</a></li>" >> body.xml
done <  $TMP_DIR/lista_tagow.txt

echo "</ul>" >> body.xml

# zapisanie listy plików dla kolejnych etapów generowania strony
 echo "multi_body=\"$out_file_list\"" >> body.info
}
```

ENGINE-PARSE_OUT.SH

```
#######################################
#         PRZETWARZANIE DO XHTML/LaTeX         #
#######################################

# funkcja odpowiedzialna za zamianę <id_link id="i:d">Opis</id_link> na linki globalne
# w dokumencie określonym przez $1 lub w standardowym wejściu
link_parser() {
 awk '
  BEGIN {
   RS="<"
  }
  /^[ \n\t]*$/ {
   next
  }

  $1 != "id_link" {
   printf("<%s", $0)
  }

  $1 == "id_link" {
   split(gensub("^id_link.*id *= *\"(.*)\".*>(.*)$", "\\1>\\2", "g"), tmp, ">")
   # powyzsza linijka robi to samo co:
```

```
  # i = match($0, "id *= *\""); if (i!=0) { $0 = substr($0, i); i = match($0, "\"") + 1 ; $0 = substr($0, i) ;
  # i = match($0, "\"") - 1 ; tmp[1] = substr($0, 0, i) ; i = match($0, ">") + 1 ; tmp[2] = substr($0, i) }
  tmp[4]=""

  for (i=0; i<9; i++) {
   # komenda okresolna przez cmd wyszukuje w bazie indeksów pasujący
   cmd=sprintf("grep \"^%s%s:*\t\" \"'"$PAGE_DIR"'/id_index\"", tmp[4], tmp[1])

   # do tmp[3] zapisujemy liczbę pasującyh indeksów, jeżeli >1 to błąd nie jest unikalny
   # jeżeli < 1 to nie odnaleziono pasującego - modyfikacja przedrostka wyszukiwania i następny obieg
   cmd2 = sprintf("%s | wc -l", cmd)
   cmd2 | getline tmp[3]
   close(cmd2)
   if (tmp[3] > 1 ) {
    printf("<a class=\"int\" href=\"[[!URL""PREFIX!]]/system.system?id=%s\">%s", tmp[1], tmp[2])
    printf("\n\033[1;31mERROR: not univocal ID ( %s ) in <id_link />\033[0m\n", tmp[1]) > "/dev/stderr"
    next
   } else if (tmp[3] < 1 ) {
    tmp[4]=sprintf("[^:]*:%s", tmp[4])
    tmp[3]=""
    continue
   } else {
    cmd2 = sprintf("%s | tr -d \"\\n\"", cmd)
    cmd2 | getline tmp[3]
    close(cmd2)
    break
   }
  }

  if (tmp[3]=="") {
   printf("\n\033[1;31mERROR: not found ID ( %s ) from <id_link />\033[0m\n", tmp[1]) > "/dev/stderr"
   printf("<a class=\"int\" href=\"[[!URL""PREFIX!]]/system.system?id=%s\">%s", tmp[1], tmp[2])
  } else {
   split(tmp[3], tmp2, "\t")
   printf("<a class=\"int\" href=\"[[!URL""PREFIX!]]%s[[!URLPOSTFIX!]]#%s\">%s", tmp2[2], tmp2[1], tmp[2])
  }
 }
' "$1" | sed \
 -e '# zamieniamy zamknięcia id_link na odpowiedni tag xhtml
  s#</id_link>#</a>#g'
}

# generacja obrazkowych równań w oparciu o ich zapis w formacie latex
parse_eq_latex() {
 awk '
  BEGIN {
   RS="<eq_latex>";
   FS="</eq_latex>";
   OFS="";
   tmp=sprintf("%s/eq_latexuation.tex", ENVIRON["TMP_DIR"])
   cmd="blahtexml --png < " tmp " | tr -d \"\n\""
```

```
  }

  NR>1&&$1!=""{
   # zapis równania do pliku tymczasowego
   unxml_eq=$1
   gsub("&lt;", "<", unxml_eq)
   gsub("&gt;", ">", unxml_eq)
   gsub("&amp;", "\\&", unxml_eq)
   print unxml_eq > tmp
   close(tmp)

   # wygenerowanie obrazka
   cmd | getline wynik
   close(cmd)

   # wygenerowanie tagu xhtml
   gsub("<blahtex><png><md5>", "", wynik)
   gsub("</md5></png></blahtex>", "", wynik)
   $1=sprintf("<img class=\"center\" src=\"./%s.png\" alt=\"$$%s$$\" />", wynik, $1)
  }
  {
   print $0
  }
 ' "$1"
}

# przetworzenie dokumentu xml określonego przez $1 na xhtml lub latex (zależnie od ⤸
wartości $OUT_MODE)
xml2view() { (
 set -e; set -o pipefail
case $OUT_MODE in
  "xhtml"*)
   { echo '<!DOCTYPE poem [
    <!ENTITY gt "&#62;">
    <!ENTITY lt "&#60;">
    <!ENTITY nbsp " ">
    <!ENTITY amp "&#38;">
    ]><document>';

    link_parser "$1";

    echo '</document>';
   } | xsltproc --param tryb "'xhtml'" --param photo_dir "'$PHOTO_DIR'" $ENGINE_DIR/base.⤸
   xsl/body.pl.src - |
    sed \
    -e '# dodanie spacji w tagach samozamykających się - kompatybilność wsteczna
     s#/>#  />#g' \
    -e '# wywalenie nagłówka xml - to tylko fragment większego dokumentu
     s#<?xml version="1.0" encoding="[UTFutf]*-8"?>##g' |
   parse_eq_latex
   # xsltproc oraz xalan są około 10 razy szybsze od saxonb-xslt, ale nie wspierają xsl 2.⤸
   0 (a ono potrzebne
   # dla latex'a (fn:replace)), ponaddto xalan zgłasza błąd gdy występuje ends-with nawet ⤸
   w nieuzywanym kodzie
   #
   # uwaga xsltproc i xalan wymagają dziwnego podawania parametrów (muszą byc ⤸
   zabezpieczone przekazanym do niego ')
   #
   # wywołanie dla xalan:
   #       xalan -xsl $ENGINE_DIR/base.xsl/body.pl.src -param tryb "'xhtml'" -param photo_dir ⤸
```

```
    "'$PHOTO_DIR'"
    ;;
   "latex"*)
    # ponadto w trybie latex ignorujemy nadmiarowe spacje w dokumencie wejsciowym
    { echo '<!DOCTYPE poem [
      <!ENTITY gt "&#62;">
      <!ENTITY lt "&#60;">
      <!ENTITY nbsp "&#126;">
      <!ENTITY amp "&#38;">
      ]><document>';

      link_parser "$1";

      echo '</document>';
    } > page.xml
    cat page.xml | sed -e 's#> <#><space /><#g' |
      saxonb-xslt -strip:all - $ENGINE_DIR/base.xsl/body.pl.src page_url=$PAGE_URL tryb=latex ⤸
      photo_dir=$PHOTO_DIR |
       sed -e 's#<?xml version="1.0" encoding="[UTFutf]*-8"?>##g'
    ;;
  esac
) }


#############################################
#          GENEROWANIE OBIEKTÓW POMOCNICZYCH         #
#############################################

# funckja wywołuje generacje obiektów (grafik, ...) tworzonych dynamicznie ze źródeł
# dla wszystkich plików źródłowych znajdujących się w katalogu zadanym przez $1
make_objects() {
  pushd "$1" > /dev/null
# generowanie pdf/png z gch
  for f in *.sch; do
   if [ -e "$f" -a ! -e "${f%.sch}.png" -a ! -e "${f%.sch}.pdf" ]; then
    suma_kontrolna=`md5sum $f | cut -f1 -d ' '`
    if ! grep "$1/$f $suma_kontrolna" "$CACHE_DIR/check_sum.txt" > /dev/null ; then
     gsch2pdf.sh $f

     echo "$1/$f $suma_kontrolna" >> "$CACHE_DIR/check_sum.txt"
     cp "${f%.sch}.png" "$CACHE_DIR/$suma_kontrolna.png"
     cp "${f%.sch}.pdf" "$CACHE_DIR/$suma_kontrolna.pdf"
    else
     ln -sr "$CACHE_DIR/$suma_kontrolna.png" "${f%.sch}.png"
     ln -sr "$CACHE_DIR/$suma_kontrolna.pdf" "${f%.sch}.pdf"
    fi
   fi
  done

# generowanie pdf/png z dia
  for f in *.dia; do
   if [ -e "$f" -a ! -e "${f%.dia}.png" -a ! -e "${f%.dia}.pdf" ]; then
    suma_kontrolna=`md5sum $f | cut -f1 -d ' '`
    if ! grep "$1/$f $suma_kontrolna" "$CACHE_DIR/check_sum.txt" > /dev/null ; then
     dia2pdf.sh $f

     echo "$1/$f $suma_kontrolna" >> "$CACHE_DIR/check_sum.txt"
     cp "${f%.dia}.png" "$CACHE_DIR/$suma_kontrolna.png"
     cp "${f%.dia}.pdf" "$CACHE_DIR/$suma_kontrolna.pdf"
    else
```

```
      ln -sr "$CACHE_DIR/$suma_kontrolna.png" "${f%.dia}.png"
      ln -sr "$CACHE_DIR/$suma_kontrolna.pdf" "${f%.dia}.pdf"
    fi
  fi
done

# generowanie pdf z pcb
for f in *.pcb; do
  if [ -e "$f" -a ! -e "$fpdf" ]; then
    suma_kontrolna=`md5sum $f | cut -f1 -d ' '`
    if ! grep "$1/$f $suma_kontrolna" "$CACHE_DIR/check_sum.txt" > /dev/null ; then
      pcb2pdf.sh $f

      echo "$1/$f $suma_kontrolna" >> "$CACHE_DIR/check_sum.txt"
      cp "$f.pdf" "$CACHE_DIR/$suma_kontrolna.pdf"
    else
      ln -sr "$CACHE_DIR/$suma_kontrolna.pdf" "$f.pdf"
    fi
  fi
done
popd > /dev/null
}
```

ENGINE-EDIT_HELP.SH

```
##################################
#       FUNKCJE - POMOC W EDYCJI       #
##################################

FUNCTIONS_HELP="$FUNCTIONS_HELP
  find_in_body() - wyszukiwanie w treści dokumentów składających się na stronę

  find_body_not_in_menu() - wyszukanie dokumentów nie przekształcanych na stronę (brak ⤸
  wpisu w menu.xml)
                          gdy jako arugument podano \"tar\" nie pokazuje plików które są ⤸
                          umieszczne tylko w archiwach tar
  find_noexist_body_in_menu() - wyszukuje wpisy w menu dla których nie istnieją odpowiednie ⤸
  pliki body.xml
  find_tar_not_in_body() - wyszukanie archiwów nie podlinkowanych na żadnej stronie
  find_notar_not_in_body() - wyszukanie rozpakowanych archiwów nie podlinkowanych na żadnej ⤸
  stronie

  create_dev_dir() - tworzy normalna strukturę źródeł programu z struktury katalogowej ⤸
  źródeł strony

  make_dir_from_source() - tworzenie struktury katalog/body.pl.src z zestawu plików
  make_menu_file_entry() - generowanie wpisów <file> pliku menu w oparciu o bieżący ⤸
  katalog
  make_menu_dir_entry() -  generowanie wpisów <dir> pliku menu w oparciu o bieżący katalog
  make_commons_link_entry() - generowanie linków do commomns w oparciu o listę katalogów z ⤸
  plikami multimedialnymi
                            (nazwy zgodne z oryginałami na commons)

  create_photo_tags() - tworzy pliki opisowe z tagami podanymi w pierwszym argumencie dla ⤸
  plików zdjęć podanych
                      jako kolejne argumenty np. create_photo_tags \"technical ups\" *.jpg

  find_too_long_lines() - wyszukuje za długie linie w plikach źródłowych
```

```
                        gdy jako arugument podano \"all\" wypisuje wszystkie zbyt długie ↩
                        linie z pliku,
                        w przeciwnym razie tylko długośc najdłuższej

  find_wiki() - wyszukanie linków do Wikipedii
  find_doc()  - wyszukanie linków \"doc\" np.:
              wyszukanie linków bez mirrorów w dziale ABC_OF_COMPUTING:
                  find_doc \"ABC_OF_COMPUTING\" | sort | grep ' >  > '
              wyszukanie linków z mirrorami w działach ABC_OF_COMPUTING i USAGE:
                  find_doc \"ABC_OF_COMPUTING USAGE\" | sort | grep -v ' >  > '

  fixWiki() - aktualizacja / edycja linków wiki (USAGE: fixWiki pl|en oldLink newLink)
  get_all_source_from_debian_repo() - pobieranie wszystkich źródeł z wskazanego ↩
  repozytorium
                                    (pierwszy argument, np. ftp://www.debian-multimedia.↩
                                    org/)
                                    w gałęzi (drugi argument - domyślnie \"sid/main\") ↩
                                    do bierzącego katalogu"

# wyszukiwanie w body.xml
find_in_body() {
 cd "$SRC_DIR"
egrep --color -r --include="body*.xml" --include="body*.src" --include="software.db.xml" ↩
"$1" .
return $?
}

# funkcja wyszukuje pliki body.xml nie występujące w zadanym pliku menu.xml
#   gdy nie okreslono pliku menu uzywa standardowego
find_body_not_in_menu() {
 cd "$SRC_DIR"
if [ "$2" = "" ]; then
  menu="$MENU_FILE"
else
  menu="$2"
fi
 find * -name "body.*" | grep -v '.svn' | grep -v 'ARCHIVE' | while read f ; do
  # dla każdego znaleźnionego pliku body.* szukamy linii w menu.xml
  # zawierającej ściezkę katalogu w którym jest ten plik
  d=`dirname "$f"`
  if ! grep "$d" "$menu" > /dev/null ; then
   dd=$d
   #while [ "$dd" != "." -a "$dd" != "" ]; do
    dd=`dirname $dd`
    [ "$1" = "tar" -a "$dd" != "." ] && grep "$dd/[*]" "$menu" > /dev/null && dd=""
   #done;
   if [ "$dd" != "" ]; then
    echo $d;
   fi
  fi
 done
}

# funkcja wyszukuje wpisy w menu.xml dla których nie istnieją odpowiednie pliki body.xml
find_noexist_body_in_menu() {
 { grep -v 'text>' OpCode_menu.xml | xmlstarlet sel -t -v '//*/dir/@path';
   grep -v 'text>' OpCode_menu.xml | xmlstarlet sel -t -v '//*/file/@path'; } | while read d; ↩
   do
  if ! ls >/dev/null 2>&1 "$d"/body.*; then
   echo "$d"
```

```
  fi
 done
}

# funkcja wyszukuje katalogi .notar nie występujące w zadanym pliku body.xml
# na podstawie wykazu z menu.xml, gdy nie okreslono pliku menu uzywa standardowego
find_tar_not_in_body() {
 cd "$SRC_DIR"
if [ "$1" = "" ]; then
  menu="$MENU_FILE"
else
  menu="$1"
fi
 awk '
 BEGIN {
  RS="<"
  FS="[=\" \t]+"
 }
 $1=="tar" {
  for (f=1; f<NF; f++) if($f=="name") {
   print $(f+1); next
  }
 }' $menu | while read f; do
  if ! find_in_body "$f.tgz" > /dev/null; then
   echo "$f -> `find_in_body "$f" | cut -f1 -d:`"
  fi
 done
}

# funkcja wyszukuje katalogi .notar nie występujące w zadanym pliku body.xml
# na podstawie wykazu z menu.xml, gdy nie okreslono pliku menu uzywa standardowego
find_notar_not_in_body() {
 cd "$SRC_DIR"
if [ "$1" = "" ]; then
  menu="$MENU_FILE"
else
  menu="$1"
fi
 awk '
 BEGIN {
  RS="<"
  FS="[=\" \t]+"
 }
 $1=="tar" && $0 ~ /show_dir[ \t]*=["'"'"']+1/ {
  for (f=1; f<NF; f++) if($f=="name") {
   print $(f+1); next
  }
 }' $menu | while read f; do
  if ! find_in_body "$f.notar" > /dev/null; then
   echo "$f -> `find_in_body "$f" | cut -f1 -d:`"
  fi
 done
}

# utworzenie katalogu zrodlowego programu z struktury katalogowej
function create_dev_dir() {
 if [ "$1" != "" ]; then
  cd "$1"
fi
```

```
if [ -d "tmp_dev_dir" ]; then
  out_name=tmp_dev_dir.`date +%s`
  echo "Moving existing tmp_dev_dir to $out_name" >&2
  mv tmp_dev_dir $out_name
 fi

if [ -e "tmp_dev_dir" ]; then
  echo "tmp_dev_dir exist and is not a directory ... exiting" >&2
  return 1;
 fi

mkdir tmp_dev_dir

function parsedir() {
  for f in $1/*; do
   basename=`basename "$f"`
   if [ "`echo $basename | cut -f1 -d.`" = "tmp_dev_dir" ]; then
    continue
   fi

   dst="tmp_dev_dir/$f"

   if [ -d "$f" -a ! -f "$f"/body.*.src ]; then
    echo "Create directory: $dst"
    mkdir "$dst"

    parsedir $f
    continue
   fi

   if [ -d "$f" ]; then
    src=`realpath "$f"/body.pl.src` # FIXME obsługa takich nie .pl.

    echo "Create link: $src -> $dst"
    ln -s "$src" "$dst"
   else
    src=`realpath "$f"`

    echo "Create link: $src -> $dst"
    ln -s "$src" "$dst"
   fi
  done
 }

parsedir .
}

# konwersja programu wieloplikowego do dir
make_dir_from_source() {
 if [ "$1" = "" ]; then
  file_list="*.h *.c *.cpp *.sh *.vhdl *.conf"
else
  file_list="$@"
fi
 for f in $file_list; do
  if [ -f $f ]; then
    mv $f body.pl.src
   mkdir $f
   mv body.pl.src $f
  fi
```

```
 done
}

# funkcja generuje wpis menu.xml z pozycjami file z bierzacego katalogu
make_menu_file_entry() {
 echo "<file level=\"\" path=\"$PWD\" />";
 ls -1Rtr -I body*.xml | grep '^[.]' | sed -e 's#^[.]/*##g' -e 's#:$##g' | while read f ; do
  echo "<file level=\"\" path=\"$PWD/$f\" />";
 done
}

# funkcja generuje wpis menu.xml z pozycjami dir -> file z bierzacego katalogu
make_menu_dir_entry() {
 echo "<file level=\"\" path=\"$PWD\" />";
 ls -1rt -I body*.xml | while read f ; do
  echo -e "<dir pdf_base_level=\"\" name=\"$f\">\n<file path=\"$PWD/$f\" />\n</dir>";
 done
}

# funcja generująca akapit z numerowanymi linkami do obrazków na commons na podstawie ⤸
katalogu z tymi obrazkami
# $1, $2, ... obrabiane podkatalogi
make_commons_link_entry() {
 echo "i=0" > /tmp/${PPID}_i
echo '<p>'
while [ -d "$1" ]; do
  . /tmp/${PPID}_i
  find ! -name '*.war' |
   grep '^./'`if [ $1 != "." ]; then echo $1; fi` |
   cut -f `if [ $1 != "." ]; then echo 3; else echo 2; fi`- -d'/' |
   while read f; do
    if [ -f "$1/$f" ]; then
     i=$(( $i+1 ))
     echo " <a class=\"ext\" href=\"http://commons.wikimedia.org/wiki/Image:$f\">[$i]</a>"
     echo "i=$i" > /tmp/${PPID}_i
    fi
   done
  shift
 done
 echo '</p>'
}

# tworzenie plików opisowych do zdjęć
create_photo_tags() {
 tagi=$1
 shift
 for f in $@; do echo -e "tags=\"$tagi\"\ndesc=\"\"" >$f.txt; done
}

find_too_long_lines() {
 linie() {
  sed -e 's#\t#____#g' -e 's#^ #_#g' -e 's#\\#\\\\#g' $1 | while read l; do
   s=`echo -n "$l" | wc -m`
   if [ $s -gt 127 ]; then
    echo -en "$s\t$1\t\t"
    echo "$l"
   fi
  done
 }
 linie_max() {
```

```
  s=`sed -e 's#\t#____#g' -e 's#^ #_#g' $1 | wc -L`
  if [ $s -gt 127 ]; then
   echo -e "$s\t$1"
  fi
 }

cd "$SRC_DIR"

if [ "$1" == "all" ]; then
  find -type f -name 'body.pl.src'| grep -v ARCHIVE | while read f; do linie     $f; done | ⤸
  sort -n
else
  find -type f -name 'body.pl.src'| grep -v ARCHIVE | while read f; do linie_max $f; done | ⤸
  sort -n
fi
}

#############################################################
#        FUNKCJE - PRZETWARZANIE ŚCIEŻEK i INNE POMOCNICZE        #
#############################################################

# konwersja identyfikatora na sciezkę do katalogu
# katalog=‘echo $identyfikator_zasobu | tr ’:’ ’/’‘
make_dir() {
 echo $1 | tr ':' '/'
}

# konwersja sciezki do katalogu na identyfikator
# identyfikator_zasobu=‘echo $katalog | tr ’/’ ’:’‘
make_id() {
 echo $1 | tr '/' ':'
}

# wykrywanie błędnych nazw katalogów
# ze wzgledu na konstrukcje identyfikatorow zasobow i jednoznaczosc
# tego przekształcenia niedozwolone sa katalogi z : w nazwie
check_dir_names() {
 if [ `find "$SRC_DIR"/[^_]* -type d | grep ':' | wc -l` != 0 ]; then
  echo -e "\\033[1;31mERROR: not allowed directory names:\\033[0m" >&2
  echo "" >&2
  find "$SRC_DIR" -type d | grep ':' >&2
  echo "" >&2
  echo -e "\\033[32mINFORMACJA: nazwy katalogów nie moga zawierać dwukropka \":\"\\033[0m" >⤸
  &2
  return 1;
 fi
 return 0;
}

# wykrywanie powtorzonych nazw plikow $1 parametr -type komendy find
check_file_names() {
 if [ "$1" = "" ]; then
  type="f"
else
  type="$1"
fi
 find "$SRC_DIR"/[^_]*/ -type $type -exec basename \{\} \; | egrep -v '^(body.*xml)|(body.⤸
 *src)|(body.info)$' |
  sort | uniq -d > "$TMP_DIR/lista_plikow"
if [ `cat "$TMP_DIR/lista_plikow" | wc -l` != "0" ] ; then
```

```
  echo -e "\\033[1;31mERROR: duplicated file names:\\033[0m" >&2
  cat "$TMP_DIR/lista_plikow" | while read nazwa_pliku ; do
   echo -e "\\033[1m$nazwa_pliku :\\033[0m" >&2
   find "$SRC_DIR"/[^_]*/ -name "$nazwa_pliku" >&2
   echo "" >&2
  done
  echo -e "\\033[32mWSKAZÓWKA: Można używać ID lub skruconego ID wraz z separatorem \":\
  \"" >&2
  echo -e "            do rozróżnienia plików z różnych katalogów\\033[0m" >&2
  return 1;
 fi
 return 0;
}

# wykrywanie niedozwolonych nazw plików
check_file_names2() {
 if [ "$1" = "" ]; then
  type="f"
else
  type="$1"
fi
 find "$SRC_DIR"/*/ -o -name page*.xhtml -o -name README*.html> "$TMP_DIR/lista_plikow"
if [ `cat "$TMP_DIR/lista_plikow" | wc -l` != "0" ] ; then
  echo -e "\\033[1;31mERROR: not allowed file names:\\033[0m" >&2
  cat "$TMP_DIR/lista_plikow"
  return 1;
 fi
 return 0;
}

# wykrywanie powtorzonych nazw sekcji <title> <head> <superhead> <minhead>
check_section_names() {
 egrep -r '(head>)|(title>)' "$SRC_DIR"/* > "$TMP_DIR/lista_plikow"
cat "$TMP_DIR/lista_plikow" | cut -d'>' -f2 | cut -d'<' -f1 | grep '[a-zA-Z0-9]' | sort |
  uniq -d > "$TMP_DIR/lista_plikow2"
if [ `cat "$TMP_DIR/lista_plikow2" | wc -l` != "0" ] ; then
  echo -e "\\033[1;31mERROR: duplicated section names:\\033[0m" >&2
  cat "$TMP_DIR/lista_plikow2" | while read nazwa_pliku ; do
   echo -e "\\033[1m$nazwa_pliku :\\033[0m" >&2
   grep "$nazwa_pliku" "$TMP_DIR/lista_plikow" | cut -f1 -d: >&2
   echo "" >&2
  done
  return 1;
 fi
 return 0;
}

# wykrywanie braku wymaganego oprogramowania
check_program() {
 if ! which $1 > /dev/null ; then
  echo -e "\\033[1;31mERROR: command \`$1\` $3 not found\\033[0m" >&2
  echo -e "\\033[32mWSKAZÓWKA: program można zainstalować poprzez:" >&2
  echo -e "            \`aptitude install ${2-1}\`\\033[0m" >&2
  return 1
 else
  return 0
 fi
}

check_programs() {
```

```
 return_code=0

check_program  gawk                                                    || return_code=1
 check_program  xsltproc                                               || return_code=1
 check_program  xmlstarlet                                             || return_code=1
 check_program  saxonb-xslt  libsaxonb-java                            || return_code=1
 check_program  highlight                                              || return_code=1
 check_program  convert                                                || return_code=1
 check_program  rsvg-convert                                           || return_code=1
 check_program  revpath      xutils-dev                                || return_code=1
 check_program  konwert                                                || return_code=1
 check_program  blahtexml                                              || return_code=1
 check_program  dvipng       dvipng             "required by blahtexml"  || return_code=1
 check_program  latex        texlive-latex-base "required by blahtexml"  || return_code=1
 check_program  gschem       geda-gschem        "required by gsch2pdf.sh" || return_code=1
 check_program  pcb          pcb                "required by pcb2pdf.sh"  || return_code=1
 check_program  dia          dia                "required by dia2pdf.sh"  || return_code=1
 check_program  ps2eps       ps2eps             "required by *2pdf.sh"    || return_code=1
 check_program  pstops       psutils            "required by *2pdf.sh"    || return_code=1
 check_program  epstopdf     texlive-font-utils "required by *2pdf.sh"    || return_code=1
 check_program  pdftoppm     poppler-utils      "required by *2pdf.sh"    || return_code=1
 check_program  pnmtopng     netpbm             "required by *2pdf.sh"    || return_code=1
 check_program  symlinks                                               || return_code=1
 check_program  debchange    devscripts                                || return_code=1

return $return_code;
}

# wyszukiwanie linkowanych artykułów Wikipedii
find_wiki() {
 cd "$SRC_DIR"
grep -r --include=body*.xml --include=software.db.xml '<wiki' ${1:-ABC_OF_COMPUTING USAGE} |
  sed -e 's#<wiki \([^>]*\)>#\n<wiki \1\n#g' | grep '^<wiki ' | cut -f2- -d' ' |
  sed -e 's#\([a-z]*\)="\([^"]*\)"#\n\1:\2\n#g' | grep -v '^ *$' | sort | uniq
}

# aktualizacja / edycja linków wiki (USAGE: fixWiki pl|en oldLink newLink)
fixWiki() {
 cd "$SRC_DIR"
grep -r --include=body*.xml --include=software.db.xml '<wiki.*="'"$2"'"' | cut -f1 -d: | ⤸
while read f; do
  sed -e 's|<wiki '"$1"'="'"$2"'"|<wiki '"$1"'="'"$3"'"|g' -i $f;
 done;
}

# wyszukiwanie linkowanych materiałów typu DOC
find_doc() {
 cd "$SRC_DIR"
grep -r --include=body*.xml --include=software.db.xml '<doc' ${1:-*} | awk 'BEGIN { RS="<⤸
 doc"; FS="</doc>" } {print $1}' |
sed -e 's#># name="#g' | awk '
  function get_tag_arg(arg, txt) {
   wynik=gensub(".*"arg"[\t ]*=[\"]([^\"]*).*", "\\1", "g", txt);
   if (wynik!=txt) return wynik
  }
  {
   printf("%s > %s > %s\n", get_tag_arg("url", $0), get_tag_arg("mirror", $0), get_tag_arg(⤸
   "name", $0))
  }
 '
```

```
}

# pobieranie wszystkich źródeł z wskazanego przez $1 repozytorium w gałęzi zadanej przez ⤸
$2
get_all_source_from_debian_repo() {
 wget $1/dists/${2-sid/main}/source/Sources.bz2
bzip2 -f -d Sources.bz2
sed -e 's#^ #CNT: #' < Sources | formail -d -s awk '
  BEGIN {FS="[: ]*"}
  $1=="CNT" && CNT==1 {print DIR "/" $4; next}
  {CNT=0}
  $1=="Directory" {DIR=$2}
  $1=="Files" {CNT=1}
 ' | while read path; do
  file=`basename $path`
  if [ ! -e $file ]; then
   wget $1/$path
  fi
 done
}
```

ENGINE-CONFIG.SH

```
# Katalog źródeł i skryptu (lub dowiązania do niego)
SRC_DIR="${SRC_DIR-/srv/WebPage/}"

# Katalog w którym będzie generowana strona
WORK_DIR="${WORK_DIR-$HOME/WebPage/}"

# Katalog w którym znajdują się pliki silnika WWW
ENGINE_DIR=$(dirname $(dirname ${BASH_SOURCE[0]}))

########################################
#       FUNKCJE KONFIGURACYJNE SILNIKA       #
########################################

# funkcja ustalająca zmienne konfiguracyjne
set_variable() {
 WWW_LANG='pl'

# tryb wyjściowy - "xhtml" lub "latex", "latex-nopdf", "latex-onepdf"
 OUT_MODE='xhtml'
# domyślny tytuł strony (dla <title />)
 STD_TITLE='OpCode - elektronika, telekomunikacja, komputery, ...'
# URL strony głównej
 PAGE_URL='http://www.opcode.eu.org/'
# opcjonalny prefiks do katalogu strony w URL
 BASE_URL_PREFIX=''
# domysne lata w notach (C)
 STD_YEAR="1999-2015"
# czas dla katalogów i plików LICENCE w plikach tar
 TAR_TIME="200609010000"
# katalog z mapą strony
 SITE_MAP_DIR="site/map"

# nazwa pliku do którego będą odnosiły się linki do katalogu
 # (puste gdy linki mają trafiać w katalog a nie odrazu w plik)
 LINK_PAGE_FILE=''
```

```
# domyślny tryb menu STATIC lub DYNAMIC (dla menu autoukrywanego)
 DEFAULT_MENU_MODE="STATIC"

# ścieżka do pliku zawierającego mapę strony
 MENU_FILE="$SRC_DIR/OpCode_menu.xml"
# skrypt parsujący mapę strony
 PARSE_MENU_AWK="$ENGINE_DIR/engine-parse_menu.awk/body.pl.src"
# skrypty konwertujące pliki pcb, sch i dia do pdf i png
 SCH2PDF="$SRC_DIR/USAGE/toolbox4electronic/software/geda_i_pcb/sztuczki_z_plikami/gsch2pdf.⤸
 sh/body.pl.src"
PCB2PDF="$SRC_DIR/USAGE/toolbox4electronic/software/geda_i_pcb/sztuczki_z_plikami/pcb2pdf.⤸
 sh/body.pl.src"
DIA2PDF="$SRC_DIR/USAGE/toolbox4electronic/software/dia/dia2pdf.sh/body.pl.src"


# katalogi z wersją wynikową oraz źródłami względem katalogu skryptu
 PAGE_DIR="$WORK_DIR/out-online"
# katalog "DOWNLOAD"
 DOWNLOAD_DIR="$WORK_DIR/out-download/"
# katalog z cache'm generowanych obrazków
 CACHE_DIR="$WORK_DIR/out-cache"
# katalog tymczasowy
 TMP_DIR="/tmp/OpCode-$USER"

# nazwa pliku z archiwum źródeł strony w dziale download (jeżeli pusta nie jest ⤸
 generowane)
 SOURCE_ARCHIVE_NAME="OpCode-source.tgz"
SOURCE_ARCHIVE_NAME=""
# linki do materiałów wyłączonych z archiwizacji źródeł umieszczane
 # na stronie działu download (względem $PAGE_DIR)
 NO_TAR_LINK=""
# ścieżki plików wylaczonych z archiwizacji zrodel (względem $SRC_DIR)
 NO_TAR="
  ARCHIVE*

  USAGE/operating_systems_config/linux/sortowanie/i18n-locales-pl_PL
 "

# nazwa pliku z archiwum off-line strony w dziale download (jeżeli pusta nie jest ⤸
 generowane)
 OFFLINE_ARCHIVE_NAME="OpCode-offline.tgz"

# lista plików tar.gz / .tgz z których budowane powinny być paczki deb
 MAKE_DEB=""

# lista plików latex z których powinny być tworzone pliki pdf
 # jedn plik w linii, po spacji można podać wymaganą liczbę powtórzeń pdflatex (⤸
 domyślnie 1)
 MAKE_PDF="
  bercik/wizytowka.tex/wizytowka.tex
  bercik/cv.tex/cv.tex
 "

# plik z okienkiem wyszukiwarki
 SEARCH_BOX="$ENGINE_DIR/_google/body.pl.xml"

# plik z bazą oprogramowania
 SOFTWARE_DB="USAGE/software_db/software.db.xml"
```

```
# katalog w drzewie wynikowym zawierający galerie i zdjęcia linkowane przez <photo_link/>
 PHOTO_DIR="/inne/photo/"


# gdy domyślny tryb konfigu usuwamy current.cfg
 if [ "$1" = "default" ]; then
  rm -f "$TMP_DIR/current.cfg"
# w przeciwnym razie włączenie tymczasowego konfigu (gdy istnieje)
 elif [ -f "$TMP_DIR/current.cfg" ]; then
  . "$TMP_DIR/current.cfg"
fi
}


# informacja o niestandardowch opcjach i działaniach skryptu
usage_conf() {
  echo "USAGE for build:" >&2
  echo "   $script_name -b" >&2
  echo "   $script_name --build" >&2
  echo
  echo "USAGE for build WWW:" >&2
  echo "   $script_name -w" >&2
  echo "   $script_name --web" >&2
  echo
  echo "USAGE for build MIRRORS dir:" >&2
  echo "   $script_name -m" >&2
  echo "   $script_name --mirrors" >&2
  echo
  echo "USAGE for build DOWNLOAD dir:" >&2
  echo "   $script_name -d" >&2
  echo "   $script_name --downloads" >&2
  echo
  echo "USAGE for build PDF:" >&2
  echo "   $script_name -p" >&2
  echo "   $script_name --pdf" >&2
  echo
  echo "USAGE for upload webpage:" >&2
  echo "   $script_name -u" >&2
  echo "   $script_name --upload" >&2
  echo

  # jeżeli jest zdeklarowana funkcja rozszerzająca działanie bierządzej to ją ⤸
  wykonyujemy
  if declare -f usage_conf2 > /dev/null; then usage_conf2; fi
}

# niestandardowe opcje i działania skryptu
run_conf() {
 # zamiana argumentów na zmienne sterujące (pozwala na podawanie argumentów w dowolnej ⤸
 kolejności)
 WEB=false; PDF=false; DOWNLOADS=false; MIRRORS=false; UPLOAD=false;
 for a in $opcje; do
  case "$a" in
   # budowanie
   "-b" | "--build")
    WEB=true; DOWNLOADS=true; PDF=true
    ;;
   "-w" | "--web")
    WEB=true
    ;;
```

```
    "-m" | "--mirrors")
     MIRRORS=true
     ;;
    "-d" | "--downloads")
     DOWNLOADS=true
     ;;
    "-p" | "--pdf")
     PDF=true
     ;;
    # upload
    "-u" | "--upload")
     UPLOAD=true
     ;;
  esac
 done

##################
 #     GENEROWANIE     #
 ##################
 if $WEB; then
  echo -e "\\033[1mGenerowanie wersji ON-LINE\\033[0m"

  set_variable default

  rm -fr "$PAGE_DIR"
  debug build_out_tree "$MENU_FILE"
fi

if $DOWNLOADS; then
  echo -e "\\033[1mGenerowanie katalogu ~DOWNLOAD\\033[0m"

  set_variable default

  # generowanie archiwum źródeł oraz paczek deb i dokumentow pdf wraz z katalogiem ⤸
  ~DOWNLOAD
  debug build_download_dir
fi

if $PDF; then
  echo -e "\\033[1mGenerowanie wersji PDF\\033[0m"

  set_variable default
  echo "OUT_MODE='latex'" > "$TMP_DIR/current.cfg"
  echo "PAGE_DIR='$WORK_DIR/LaTeX-PDF'" >> "$TMP_DIR/current.cfg"
  . "$TMP_DIR/current.cfg"

  rm -fr "$PAGE_DIR"
  debug build_out_tree "$MENU_FILE"

  # przeniesienie do ~DOWNLOAD
  cp -f "$PAGE_DIR/page.pdf" "$DOWNLOAD_DIR/OpCode-book.pdf"
fi

#############
 #     UPLOAD     #
 #############
 if $UPLOAD; then
  echo -e "\\033[1mUpload przygotowanych wcześniej wersji ON-LINE (dir), OFF-LINE (tar), PDF ⤸
  (pdf)\\033[0m"
```

```
    set_variable default

    # upload_web_page
    cd "$PAGE_DIR"
    rsync -rLc --delete --exclude="~rrp" -v -e "ssh -p 22 -l rrp" . www.opcode.eu.org:⤸
    /srv/SERVICES/www/site

    # generate_sitemap
    ssh rrp@www.opcode.eu.org 'cd /srv/SERVICES/www/site;
     find -type f -exec chmod 644 {} +; find -type d -exec chmod 755 {} +;
     bash usage_and_config/toolbox4computer/www_engine/generate_google_sitemap.⤸
     sh/generate_google_sitemap.sh > sitemap.xml'
fi

# jeżeli jest zdeklarowana funkcja rozszerzająca działanie bierządzej to ją wykonyujemy
 if declare -f run_conf2 > /dev/null; then run_conf2; fi
}

# Opcje dla funkcji zarzadzania repozytoriami
repo_do_case() {
 case $* in
 "H hostname1")
  push_$ACTION "up-host" "/up-path/$REPO_PATH" "$LOCAL_HOSTNAME" "$LOCAL_REPOPATH"
  ;;
 "HA hostname1 alias1")
  if [ "$ACTION_DIR" != "from" ]; then
   # hostname1-spec-cmd
   push_$ACTION "alias-up-host" "/up-path/$REPO_PATH" "$LOCAL_HOSTNAME" "$LOCAL_REPOPATH"
  fi
  ;;
 "A alias1")
  push_$ACTION "alias-up-host" "/up-path/$REPO_PATH" "$LOCAL_HOSTNAME" "$LOCAL_REPOPATH"
  ;;
 *)
  return 2
  ;;
 esac
 return 0
}



#################################################
#       ZMIENNE NAPISOWE WYKORZYSTYWANE W SILNIKU        #
#################################################

# domyślny posiadacz praw autorskich, typ licencji itp.
COPYRIGHT_LINE="Robert Paciorek (http://www.opcode.eu.org/), BSD/MIT-type license"
COPYRIGHT_LINE_XHTML="Robert Paciorek (http://www.opcode.eu.org/), <a class=\"Wikipedia\" ⤸
href=\"http://pl.wikipedia.org/wiki/Licencja_BSD\">BSD</a>/<a class=\"Wikipedia\" href=\"http:⤸
//pl.wikipedia.org/wiki/Licencja_X11\">MIT</a>-type license"
COPYRIGHT_LINE_LATEX="Robert Paciorek (\url{http://www.opcode.eu.org/}), \href{http://pl.⤸
wikipedia.org/wiki/Licencja_BSD}{BSD}\footnote{http://pl.wikipedia.org/wiki/Licencja\_BSD}⤸
/\href{http://pl.wikipedia.org/wiki/Licencja\_X11}{MIT}\footnote{http://pl.wikipedia.⤸
org/wiki/Licencja\_X11}-type license"

# licencja PL i EN
LICENCE_1="Redystrybucja wersji źródłowych i wynikowych, po lub bez dokonywania
modyfikacji JEST DOZWOLONA, pod warunkiem zachowania niniejszej informacji o
prawach autorskich. Autor NIE ponosi JAKIEJKOLWIEK odpowiedzialności za skutki
```

```
użytkowania tego dokumentu/programu oraz za wykorzystanie zawartych tu informacji."
LICENCE_2="This text/program is free document/software. Redistribution and use in
source and binary forms, with or without modification, ARE PERMITTED provided
save this copyright notice. This document/program is distributed WITHOUT any
warranty, use at YOUR own risk."

# dodatkowe info odnośnie licencji dołączane do każdego głównego pliku lICENCE w ⤸
archiwach TAR
TAR_COPYRIGHT_INTRO="Jeżeli jawnie nie podano inaczej wszystkie pliki w archiwum
dystrybuowane są na następującej licencji:\n
All file in this archive (except when copyright in file say otherwise)
is distributed on following licence:\n\n"

# funkcja wypisująca dodatkowe info w stopce XHTML katalogow notar
# argumenty:
#    $1 = "URL do pliku tar",
notar_copyright_intro() {
 echo "<p>Strona jest częścią serwisu <a href=\"$PAGE_URL\">OpCode</a>. Strona prezentuje ⤸
 rozpakowaną wersję archiwum dostępnego pod adresem <a href=\"$1\">$1</a>. Po szczegóły ⤸
 dotyczące prezentowanego materiału zapraszam do katalogów nadrzędnych.</p>"
echo "</p><p>Jeżeli jawnie nie podano inaczej wszystkie pliki w archiwum dystrybuowane są ⤸
 na następującej licencji:<br />
 All file in this archive (except when copyright in file say otherwise) is distributed on ⤸
 following licence:</p>"
}

# funkcja wypisująca dodatkowe info w stopce XHTML
# argumenty:
#    $1 = "URL dokumentu",
#    $2 = "URL strony",
#    $3 = "data modyfikacji",
#    $4 = "prefix URL strony"
xhtml_fotter_addons() {
 echo '<p class="fotter"><a href="http://validator.w3.org/check/referer">
 <img src="/valid-xhtml11.png" alt="Valid XHTML 1.1" class="right noprint" height="31" ⤸
 width="88" /></a>
 Dokument ten (URL: '$1')
 należy do serwisu <a href="'$2'">OpCode</a>. Autorem tej strony jest <a href="'$4'/bercik/">⤸
 Robert Paciorek</a>,
 wszelkie uwagi proszę kierować na adres e-mail serwisu: <a href="mailto:webmaster@opcode.⤸
 eu.org">webmaster@opcode.eu.org</a>.
 <br />Data ostatniej modyfikacji artykulu: '$3' (data ta może być zafałszowana ⤸
 niemerytorycznymi modyfikacjami artykułu).</p>'
}

# informacja na stronie DOWNLOAD dodawana odpowiednio przed i po liście $NO_TAR_LINK (⤸
jeżeli $DOWNLOAD_README_1 puste lub nie ustawione nie jest generowane README.xhtml)
#DOWNLOAD_README_1="<p>Archiwum <a href=\"./OpCode-source.tgz\">OpCode-source.tgz</a> zawiera ⤸
całość źródeł strony poza <a href=\"/archiwum/\">archiwum</a>, galeriami <a ⤸
href=\"/photo/\">zdjęć</a> i działem DOWNLOAD (pliki nie stanowiące integralnego elementu ⤸
mojej strony) oraz następującymi materiałami których nie jestem wyłącznym autorem (są ⤸
to głównie modyfikacje istniejących programów oraz plików konfiguracyjnych/skryptów ⤸
Debiana; pliki te są dystrybuowane na innych warunkach niż całość serwisu):</p>"
#DOWNLOAD_README_2="<p>Wszystkie pliki serwisu mojego autorstwa, zamieszczone w archiwum ⤸
dystrybuowane są na zasadach licencji typu BSD, chyba że wyraźnie zastrzeżono inaczej.</p>⤸
<p>Archiwum <a href=\"./OpCode-offline.tgz\">OpCode-offline.tgz</a> zawiera całość serwisu ⤸
(poza <a href=\"/archiwum/\">archiwum</a>, galeriami <a href=\"/photo/\">zdjęć</a> i ⤸
działem DOWNLOAD) do przeglądania w trybie off-line. Plik <a href=\"./OpCode-book.pdf\">⤸
OpCode-book.pdf</a> zawiera \"książkową\" (format A4) wersję wybranych ⤸
artykułów/materiałów z serwisu.</p>"
```

```
####################################################################
#        SZABLONY TEKSTOWE WYKORZYSTYWANE W SILNIKU (FUNKCJE BASH)       #
####################################################################

# funkcja komentuje każdą linijkę podanego tekstu ($2) zadanym napisem ($1)
comments_line() {
 echo -e "$2" | while read l; do
  echo "$1 $l";
 done
}

# wypisuje standardową informacje o prawach autorskich i licencji
# $1 okresla typ wypisywania, $2 podaje informacje o dacie, $3   współautorzy
#   $4 (opcjonalny) - dodatkowe informacje Copyright
std_copyright() {
 case $1 in
  c | cpp | php | h | css | js)
   # komentarze w stylu C
   echo -e "/**"
   comments_line " *" "Copyright (c) $2, $COPYRIGHT_LINE"
   comments_line " *" "$3"
   echo -e " *"
   comments_line " *" "$LICENCE_1"
   echo -e " *"
   comments_line " *" "$LICENCE_2"
   comments_line " *" "$4"
   echo -e "**/\n"
   ;;
  tex | latex)
   # komentarze TeX/LaTeX
   comments_line "%" "Copyright (c) $2, $COPYRIGHT_LINE"
   comments_line "%" "$3"
   echo -e "%"
   comments_line "%" "$LICENCE_1"
   echo -e "%"
   comments_line "%" "$LICENCE_2"
   comments_line "%" "$4"
   ;;
  ini)
   # komentarze plików typu ini (np. konf. Asterisk'a)
   comments_line ";" "Copyright (c) $2, $COPYRIGHT_LINE"
   comments_line ";" "$3"
   echo -e ";"
   comments_line ";" "$LICENCE_1"
   echo -e ";"
   comments_line ";" "$LICENCE_2"
   comments_line ";" "$4"
   ;;
  lua|vhd|vhdl)
   # komentarze plików typu ini (np. konf. Asterisk'a)
   comments_line "--" "Copyright (c) $2, $COPYRIGHT_LINE"
   comments_line "--" "$3"
   echo -e "--"
   comments_line "--" "$LICENCE_1"
   echo -e "--"
   comments_line "--" "$LICENCE_2"
   comments_line "--" "$4"
```

```
    ;;
  htm | html | xhtml)
    # kodowanie html
    echo -e "Copyright (c) $2, $COPYRIGHT_LINE_XHTML
     <br />$3<br /><br />$LICENCE_1<br /><br />$LICENCE_2<br />$4"
    ;;
  latex-nocomment)
    # kodowanie TeX/LaTeX
    echo '\chapter{Copyright}'
    echo "\noindent Copyright (c) $2, $COPYRIGHT_LINE_LATEX\\"
    echo "$3\\ \\"
    echo "$LICENCE_1\\ \\"
    echo "$LICENCE_2\\ \\"
    echo "$4"
    ;;
  txt)
    # bez komentarzy
    echo -e "Copyright (c) $2, $COPYRIGHT_LINE"
    echo -e "$3"
    echo -e ""
    echo -e "$LICENCE_1"
    echo -e ""
    echo -e "$LICENCE_2"
    echo -e ""
    echo -e "$4"
    ;;
  *)
    # komentarze shellowe
    comments_line "#" "Copyright (c) $2, $COPYRIGHT_LINE"
    comments_line "#" "$3"
    echo -e "#"
    comments_line "#" "$LICENCE_1"
    echo -e "#"
    comments_line "#" "$LICENCE_2"
    comments_line "#" "$4"
    ;;
 esac
}

# wypisuje niestandardową informacje o prawach autorskich i licencji
# $1 okresla typ wypisywania, $2 niestandardowe informacje Copyright
special_copyright() {
 case $1 in
  c | cpp | php | h | css | js)
    # komentarze w stylu C
    echo -e "/**"
    comments_line " *" "$2"
    echo -e "**/\n"
    ;;
  tex | latex)
    # komentarze TeX/LaTeX
    comments_line "%" "$2"
    ;;
  ini)
    # komentarze plików typu ini (np. konf. Asterisk'a)
    comments_line ";" "$2"
    ;;
  lua|vhd|vhdl)
    # komentarze plików typu ini (np. konf. Asterisk'a)
    comments_line "--" "$2"
```

```
    ;;
   htm | html | xhtml)
    # kodowanie html
    echo -e "$4"
    ;;
   latex-nocomment)
    # kodowanie TeX/LaTeX
    echo '\chapter{Copyright}'
    echo "\noindent $4"
    ;;
   txt)
    # bez komentarzy
    echo -e "$4"
    ;;
   *)
    # komentarze shellowe
    comments_line "#" "$4"
    ;;
  esac
}

# funkcja generuje nagłówk XHTML
# $1 - tytyl $2 - prefix url dla włączanych plików css $3 - layout mode $4 - inhead
std_xhtml_header() {
 echo '<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.
dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pl">
<head>
 <meta http-equiv="ContentType" content="application/xhtml+xml; charset=utf-8" />
 <title>' "$1" '</title>
 <link rel="stylesheet" href="'"$2"'/base.css" type="text/css" />
 <link rel="stylesheet" href="'"$2"'/layout_'"$3"'.css" type="text/css" />
 <link rel="stylesheet" href="'"$2"'/highlight.css" type="text/css" />
 <link rel="home" title="Strona główna" href="'$PAGE_URL'" />
 '$4'
</head><body><div class="tresc">'
echo -e "\n";
}

# funkcja generuje stopkę XHTML
# $1 - sciezka do dokumentu $2 - data aktualizacji $3 - menu $4 - nagłówek menu
# $5 - prefix url dla linków itp.
# $6 - lata dla Copyright $7 - współautorzy dla Copyright
# $8 (opcjonalne) - dodatkowe informacje Copyright w formie xhtml
std_xhtml_fotter() {
 echo -e "\n<hr /></div>"
if [ "$3" != "no" ] ; then
  echo -e "\n<div class=\"menu\"><p class=\"menu_info\">M<br />E<br />N<br />U</p>\n\n";
  echo "$4"
  echo '<ul class="menu_link">'
  cat "$3"
  echo '</ul>'
  if [ $OUT_MODE = "xhtml" -a $SEARCH_BOX != "" ]; then
   cat $SEARCH_BOX
  fi
  echo -e "\n</div>";
 fi

echo -e '\n<div class="tresc"><hr />\n<p class="fotter">'
```

```
std_copyright htm "$6" "$7" "$8"
echo '</p>'
xhtml_fotter_addons \
  "$PAGE_URL`echo "$1" | awk -v FS="$PAGE_DIR" '{if ($2!="") print $2; else print $1 }'`" \
  "$PAGE_URL" "$2" "$5"
echo '</div>'

echo -e '\n\n</body></html>'
}


std_latex_header() {
 echo '\documentclass[a4paper, portrait, 110pt]{mwrep}
 \usepackage{fontspec,polski}
 \usepackage[a4paper,tmargin=2.2cm,bmargin=2.2cm,lmargin=2.2cm,rmargin=2.2cm]{geometry}
 \usepackage[unicode=true,pdfpagemode=UseOutlines,colorlinks=true,linkcolor=black,⤸
 urlcolor=blue]{hyperref}
 \usepackage{graphics,color,floatflt,wrapfig,adjustbox}
 \usepackage{fancyhdr,lastpage,url}
 \usepackage[iso]{isodateo}
 \usepackage{scalefnt,relsize,soul,soulutf8,ulem}
 \usepackage{amsmath,amssymb}
 \usepackage{array,longtable,tabu,multirow,hhline}
 \usepackage[final=false]{minted}

 \pdfimageresolution=150
 \DeclareGraphicsExtensions{.pdf,.mps,.png,.jpg}
 \hypersetup{pdfauthor={Robert Paciorek, http://www.opcode.eu.org/},pdftitle={Komputerologia ⤸
 dla każdego}}
 \setmainfont[BoldFont=* Bold]{Linux Libertine O}\defaultfontfeatures{Ligatures=TeX}
 \setminted{breaklines=true, breakanywhere=true, breakafter=-/.()}

 % obsługa kolorowanego kodu
 \usepackage{marvosym}
 \input{highlight.sty}

 % przypisy na koncu dokumentu bez własnego tytułu
 \usepackage{endnotes} \let\footnote=\endnote \renewcommand{\notesname}{}

 % wszystko (także subsubsection) w spisie treści, brak numeracji
 \setcounter{tocdepth}{6}\setcounter{secnumdepth}{-2}

 % korekta formatowania nagłówków
 \SetSectionFormatting{part}          {2.8em} {\centering\Huge\textbf{\HeadingText}}  {2em}
 \SetSectionFormatting{chapter}       {2.5em} {\Huge\textbf{\textsc{\HeadingText}}}  {1.6em}
 \SetSectionFormatting{section}       {2.2em} {\huge\textbf{\textsc{\HeadingText}}}  {.9em}
 \SetSectionFormatting{subsection}    {1.9em} {\LARGE\textbf{\textsc{\HeadingText}}} {.8em}
 \SetSectionFormatting{subsubsection} {1.6em} {\Large\textbf{\textsc{\HeadingText}}} {.7em}
 \SetSectionFormatting{paragraph}     {1.3em} {\large\textbf{\textsc{\HeadingText}}} {.6em}
 \SetSectionFormatting{subparagraph}  {1em}   {\textbf{\HeadingText}}                {.5em}

 \sloppy

 % strona tytułowa, spis tresci, ...
 \begin{document}
  \thispagestyle{empty}
  \vspace*{5cm}
  \begin{center}{\Large \bf Robert Paciorek}\end{center}
  \vspace*{1.5cm}
  \begin{center}{\scalefont{3.3} \bf Komputerologia dla każdego}\end{center}
```

```
  \vspace*{0.5cm}
  \begin{center}{\large \url{http://www.opcode.eu.org/}}\end{center}
  \vspace*{8cm}
  \begin{center}\today\end{center}
  \pagebreak

  \tableofcontents\newpage

  \setlength{\parindent}{3em}
'
}

std_latex_fotter() {
 std_copyright latex-nocomment "$1" "$2" "$3"
echo "\chapter{Przypisy}\theendnotes\end{document}"
}



#############################################
#       WCZYTANIE POZOSTAŁYCH PLIKÓW SILNIKA        #
#############################################
. $ENGINE_DIR/engine-generate.sh/body.pl.src
. $ENGINE_DIR/engine-parse_in.sh/body.pl.src
. $ENGINE_DIR/engine-parse_out.sh/body.pl.src
. $ENGINE_DIR/engine-edit_help.sh/body.pl.src
. $ENGINE_DIR/engine-base.sh/body.pl.src
```

SYSTEM.PHP

```
<?php

//   KONFIGURACJA HTTPD na potrzeby użytkowania tego skryptu
//   umieszczonego pod nazwą system.system
//
//   # konfiguracja vhosta
//   <VirtualHost *:80>
//     DirectoryIndex page.xhtml
//
//     RewriteEngine On
//
//     # obsługa skróconej nazwy dla system.system
//     RewriteRule ^/sys[?] /system.system [L]
//     RewriteRule ^/sys$ /system.system [L]
//
//     # XHTML tylko dla Mozilla - rozwiązanie problemu z GoogleBot oraz IE
//     AddType text/html;charset=utf-8 .xhtml
//     BrowserMatch "Mozilla/5" xhtml_head=yes
//     BrowserMatch "Mozilla/6" xhtml_head=yes
//     RewriteCond %{ENV:xhtml_head} =yes
//
//     # nazwa domeny z www.
//     RewriteCond %{HTTP_HOST} !=www.opcode.eu.org
//     RewriteRule ^/(.*)$ http://www.opcode.eu.org/$1 [R=301,L]
//
//     # zabezpieczenie przed wywołaniem pliku page.xhtml lub README.html
//     RewriteRule ^([^~]*)/page.xhtml$ $1 [R=301,L]
```

```
//     RewriteRule ^([^~]*)/README.xhtml$ $1 [R=301,L]
//   </VirtualHost>
//
//   # konfiguracja dla katalogu na którym operuje vhost oraz odwołania poza-vhostowe
//   <Directory ~ /home/rrp/public_html[^_].*>
//     RewriteEngine On
//     # gdy nie jest to moj vhost oraz istnieje plik page.xhtml w wskazywanym katalogu
//     # to korzystamy z system.system
//     RewriteCond %{DOCUMENT_ROOT} !/home/rrp/public_html
//     RewriteCond "%{REQUEST_FILENAME}/page.xhtml" -f
//     RewriteRule ^[^~]*/$ /~rrp/system.system [L]
//     # uwaga: warunek ^[^~]*/$ implikuje że jest to plik domyślny katalogu
//     #              wygenerowany przez mój silnik, a więc xhtml ...
//
//     # obsługa skróconej nazwy dla system.system
//     RewriteRule ^/~rrp/sys /~rrp/system.system [L]
//
//     # automatyczne indeksy
//     IndexOptions XHTML HTMLTable Charset=utf-8
//     ReadmeName README.xhtml
//     IndexIgnore README.xhtml
//
//     # .php traktujemy jako zwykly tekst a .system jako php
//     AddType text/plain .php
//     AddType application/x-httpd-php .system
//   </Directory>


/// obsługa (dynamiczne generowanie) pliku robots.txt
if ( $_GET["robots"] == "old_site" ) {
 header("Content-Type: text/plain", 1);
 echo "User-agent: ia_archiver\nDisallow:\n\n";
 echo "User-agent: Googlebot\nDisallow: /\n\n";
 echo "User-agent: *\nDisallow: /archiwum/\n\n";
 exit;
}



/******************************
 * PRZETWARZANIE ŚCIEŻEK i URL *
 ******************************/

/// wyrzucamy pytajniki
$tmp = explode("?", urldecode($_SERVER['REQUEST_URI']));
$REQUEST_URL = $tmp[0];


/// wybieramy wariant serwera - vhost lub ~user
if ($REQUEST_URL[1] != "~") {
 $BASE_PATH = $_SERVER['DOCUMENT_ROOT'];
 $BASE_URL = "";

/// wycinamy rozpoczynający system.system lub sys (jeżeli był)
 if ( substr($REQUEST_URL, 0, 14) == "/system.system") {
  $url_args = "system.system";
  $REQUEST_URL = substr($REQUEST_URL, 14);
 } else if ( substr($REQUEST_URL, 0, 4) == "/sys") {
  $url_args = "sys";
  $REQUEST_URL = substr($REQUEST_URL, 4);
```

```
  }
} else {
  $BASE_PATH = "/home/rrp/public_html";
  $BASE_URL = "/~rrp";
  $url_args = "";

/// wycinamy rozpoczynające ~rrp
  $tmp = explode("~rrp", $REQUEST_URL);
  $REQUEST_URL=$tmp[1];
}

/// ustalenie $URL_POSFTIX w oparciu o argumenty rządania (id i page - obsługiwane poniżej)
$URL_POSFTIX = "?";
foreach(array("no_pl", "no_redirect", "static_menu") as $arg) {
  if (isset($_GET[$arg]))
    $URL_POSFTIX .= $arg . "&amp;";
}

/// obsługujemy id_link (id) i page
if (isset($_GET["id"])) {
  $request_id=$_GET["id"];
  $tmp[0]="";
  for ($i=0; $i<9; $i++) {
    # komenda okresolna przez cmd wyszukuje w bazie indeksów pasujący
    $cmd=sprintf("grep \"^%s%s:*\t\" id_index", $tmp[0], $request_id);

    exec ($cmd, $tmp2);

    if (count($tmp2) < 1 ) {
      $tmp[0]=sprintf("[^:]*:%s", $tmp[0]);
      continue;
    } else if (count($tmp2) > 1 ) {
      $REQUEST_URL="not_found";
      break;
    } else {
      $tmp=explode("\t", $tmp2[0]);
      $tmp=explode("#", $tmp[1]);
      $REQUEST_URL=$tmp[0];
      break;
    }
  }
  if ( ! isset($_GET["no_redirect"]) && ! isset($_GET["static_menu"])
  ) ) {
    # gdy nie zakazano redirectu to go robimy
    $REQUEST_URL = $_SERVER['HTTP_HOST'] . $BASE_URL . $REQUEST_URL . "#" . $request_id;
    header("Location: http://" . ereg_replace("//+", "/", $REQUEST_URL), 1);
    exit;
  }

/// FIXME to powinno być tak: $URL_POSFTIX .= "id=";
  $URL_POSFTIX = ""; $url_args = "";
} else if ($url_args != "") {
// domyslne dla linkowania gdy wywolano z vhosta bez id
//   takze gdy wywolano jawnie z page -> if (isset($_GET["page"]))
  $tmp=explode("#", $_GET["page"]);
  $REQUEST_URL=$tmp[0];

$URL_POSFTIX .= "page=";
}
```

```
/// korekcja $BASE_URL i $URL_POSFTIX w oparciu o $url_args
if ($url_args != "") {
  // było jawne wywołanie system.system o postaci podanej w $URL_ARGS
  // zatem argumenty URL dodajemy do URL_BASE zamiast korzystać z URL_POSTFIX
  $BASE_URL = $url_args . $URL_POSFTIX;
  $URL_POSFTIX = "";
}

/** w tym miejscu mamy ustawione poprawne:
        $REQUEST_URL - wskazuje na ścieżkę rządanego zasobu na serwerze względem $BASE_PATH
        $BASE_URL - uzupełnienie url z lewej strony
        $URL_POSFTIX - uzupełnienie url z prawej strony
        $BASE_PATH - uzupełnienie ścieżki do pliku na serwerze **/



/***********************************************
 * PRZETWARZANIE I WYSYŁANIE TREŚCI DOKUMENTU *
 ***********************************************/

/// obsługujemy usuwanie PL znaczków i zmianę nagłóka na ASCII
if (isset($_GET["no_pl"])) {
  /* $WITHOUT_PL = ' | awk \'
 BEGIN {
    LENG=split("ą ę ś ć ż ó ź ń Ą Ę Ś Ć Ż Ó Ź Ń ł Ł", IN, " ");
    split("a e s c z o z n A E S C Z O Z N l L", OUT, " ");
 }
 {
    for (i=1; i<=LENG; i++) {
      gsub(IN[i], OUT[i]);
    };
    print $0;
 }\"; */
$WITHOUT_PL = ' | konwert UTF8-ascii';
  $CONTENT_TYPE="Content-Type: text/html; charset=ASCII";
} else {
  $WITHOUT_PL = '';
  $CONTENT_TYPE="Content-Type: text/html; charset=UTF-8";
}

/// tworzymy pelną scieżkę do pliku ($FILE_PATH) w oparciu o $BASE_PATH i $REQUEST_URL
/// w tym gdy nie zawiera określenia dokumentu to je dodajemy
/// chyba że jest istniejącym plikiem (obrazek, etc) - wtedy wysyłamy
$FILE_PATH = $BASE_PATH . $REQUEST_URL;
$tmp = explode("/", urldecode($_SERVER['REQUEST_URI']));
if ( $tmp[count($tmp)-1] != "page.xml" ) {
  if ( is_file("$FILE_PATH") && strstr(realpath("$FILE_PATH"), $BASE_PATH) != "" ) {
    $cmd = popen("/usr/bin/file -bi '$FILE_PATH'", 'r');
    $CONTENT_TYPE = "Content-Type: ";
    $CONTENT_TYPE .= fread($cmd, 2096);
    pclose($cmd);
    header($CONTENT_TYPE, 1);
    system("cat '$FILE_PATH'");
    exit;
  } else if ( is_file("$FILE_PATH/page.xhtml") ) {
    $FILE_PATH .= "/page.xhtml";
  } else if (is_dir("$FILE_PATH")) {
    // dla indeksowanych automatycznie katalogów
    // w trybie v-hostowym wypisujemy katalog i kończymy
    // w trybie nie v-hostowym nie powinien być wywoływany
```

```
  $listing=file_get_contents("http://" . $_SERVER['SERVER_NAME'] . $REQUEST_URL);
  echo str_replace("href=\"", "href=\"" . $BASE_URL . $REQUEST_URL, $listing);
  exit;
 }
}

// finalna korekcja zmiennych $BASE_URL, $URL_POSFTIX, $REQUEST_URL
// tak aby zabezpieczyć je przed sed'em
$BASE_URL     = str_replace('&', '\&', $BASE_URL);
$URL_POSFTIX = str_replace('&', '\&', $URL_POSFTIX);
$REQUEST_URL = str_replace('&', '\&', $REQUEST_URL);

/// obsługujemy alternatywne arkusze CSS
if (isset($_GET["static_menu"])) {
 $MOD_CSS = ' -e \'s#<link[^>]*layout_dynamic.css[^>]*>#<link rel="stylesheet" href="' .
  $BASE_URL . '/layout_static.css" type="text/css" />#g\'';
} else {
 $MOD_CSS = '';
}

/// gdy istnieje wypisujemy stosowny dokument wraz z modyfikacjami:
/// * linki globalne poprzedzone $BASE_URL
/// * meta name="robots"
/// * PL znaczki
header($CONTENT_TYPE, 1);
if ( is_file("$FILE_PATH") && strstr(realpath("$FILE_PATH"), $BASE_PATH) != "" ) {
 system ('sed ' .
  // brak indeksowania przez roboty (indeksujemy tylko normalną stronę)
  ' -e \'s#<title>#<meta name="robots" content="noindex, nofollow" /><title>#g\'' .
  $MOD_CSS .
  // linki globalne "/COS" (dwa warianty - gdy mamy # i gdy go nie ma)
  ' -e \'s%<\([^<]*\)"/\([^>"]*\)#\([^>"]*\)"%<\1"' . $BASE_URL . '/\2' . $URL_POSFTIX . ↩
  '#\3"%g\' ' .
  ' -e \'s%<\([^<]*\)"/\([^>#"]*\)"%<\1"' . $BASE_URL . '/\2' . $URL_POSFTIX . '"%g\' ' .
  // linki lokalne "./COS"
  ' -e \'s#<\([^<]*\)"./\([^>"]*\)"#<\1"' . $BASE_URL . $REQUEST_URL . '/\2"#g\' ' .
  // brak PL znaczków (jeżeli był żądanay)
  $FILE_PATH .
  $WITHOUT_PL);
 exit;
}


/// gdy nie istnieje wypisujemy stronę błędu 404
header("Content-Type: application/xhtml+xml; charset=utf-8", 1);
header("HTTP/1.0 404 Not Found", 1);
header("Status: 404 Not Found", 1);
echo "<?xml version=\"1.0\" encoding=\"utf-8\" ?>";
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.↩
dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pl">
<head>
 <title>HTTP 404 error (www.opcode.eu.org)</title>
 <link rel="home" title="Strona główna" href="http://www.opcode.eu.org/" />
</head><body>

<h1>HTTP 404 error (Not Found)</h1>

<p>Strona o podanym adresie URL nie istnieje</p>
```

```
<p>Zapraszamy na <a href="http://www.opcode.eu.org/">stronę główną</a>.</p>

</body></html>
```

BASE.CSS

```css
/** STANDARDOWY TEKST I TŁO **/
p {
 text-align:justify;
 text-indent: 25pt;
}
p.noindent {
 text-indent: 0;
}
body {
 background-color: #f0f8ff;
}

/** WYROWNANIA **/
*.center {
 text-align: center;
}
table.center {
 margin-left: auto;
 margin-right: auto;
}
.right {
 float: right;
 padding: 5px;
}
.left {
 float: left;
 padding: 5px;
}

/** OBRAZKI **/
img {
 border: 0px none;
}

/** ZNACZNIKI LOGICZNE **/
dfn, cite, var, samp, blockquote {
 font-style: italic;
}
em {
 font-weight: bold;
}
pre {
 background-color: #E1FDF0;
}
code {
 background-color: #EBFFF5;
}

/** WYRÓŻNIENIA **/
.important-app {
 font-weight: bold;
}
```

```css
/** LINKI **/
a.int, span.int {
 /* text-decoration:none ; */
color: blue;
}
a.Wikipedia, span.Wikipedia {
 /* text-decoration:none ; */
color: green;
}
a.ext_doc, a.ext_doc_site, span.doc {
 /* text-decoration:none ; */
color: teal;
}
a.ext, a.ext_homepage, span.ext {
 /* text-decoration:none ; */
color: navy;
}
a[class]:visited, span.visited {
 /* text-decoration:none ; */
background-color: #D6D6D6;
}
a[class]:focus, a[class]:hover {
 text-decoration:underline;
 color: red;
}
span.linkWithMirror a.mirror {
 display: none;
}
span.linkWithMirror:hover a.mirror {
 display: inline-block;
 vertical-align: middle;
 position: relative;
 top: -2px;

width: 4em;
 font-size: 60%;
 text-align: center;
 text-indent: 0;
 padding-left: 0.7em;
}

/** AKAPITY SPECJALNE **/
p.download , a.download {
 font-size : large;
 font-weight : bold;
 text-align: center;
 text-indent: 0;
}
p.fotter {
 font-size: xx-small;
 text-indent: 0;
}

p.art_menu {
 font-style: italic;
 font-size: small;
 text-align: center;
 text-indent: 0;
}
```

```
a.art_menu0, a.art_menu1 {
 font-weight : bold;
}

a.art_menu3 {
 font-style: italic;
}

p.menu_path {
 font-size: small;
 font-weight: bold;
 text-align: center;
 text-indent: 0;
 padding-bottom: 5pt;
 border-bottom: groove 4px #e3e3e3;
}
ul.menu_link {
 font-weight: bold;
 text-align: left;
 text-indent: 0;
 list-style-type: none;
 padding: 0;
}
ul.menu_link li {
 padding: 5pt;
}
ul.submenu_link, ul.subsubmenu_link {
 font-weight: bold;
 text-align: left;
 text-indent: 0;
 list-style-type: none;
 padding: 0;
}
ul.submenu_link li {
 font-weight: normal;
 padding: 4pt 2pt 4pt 10pt;
}
ul.subsubmenu_link li {
 font-weight: normal;
 padding: 3pt 2pt 3pt 15pt;
}
ul.submenu_link li:before, ul.subsubmenu_link li:before {
 font-weight: normal;
 margin-right: 7pt;
 content: "→";
}
li.group_head {
 font-weight: bold;
 text-align: center;
}
dl {
 margin-left: 50pt;
}

form.search {
 background-color: #e3e3e3 ;
}
form.search p {
 text-indent: 0;
```

```
 padding: 2pt;
}
form.search p {
 font-size: small;
 text-align: center;
}
form.search p#searchInfo {
 font-size: x-small;
 text-align:justify;
}
form.search p label#searchTitle {
 font-size: medium;
}
form.search #q {
 width: 90%;
}

/** DLA WYDRUKU - LINKI **/
@media print {
 body {
  background-color: white;
 }

a[class] {
  color: black;
  text-decoration: none;
 }

a.ext_doc:after, a.ext_doc_site:after, a.Wikipedia:after, a.ext:after, a.ext_homepage:after ⤸
{
  font-style: italic;
  content: " [" attr(href) "]";
 }

a.int:after {
  font-style: italic;
  content: " [http://www.opcode.eu.org" attr(href) "]";
 }

a:visited {
  background-color: #FFFFFF;
 }

p.art_menu, .noprint {
  display: none;
  visibility: hidden;
 }

pre {
  white-space: pre-wrap;
 }
}

/** AUTOMATYCZNE SKALOWANIE OBRAZKÓW Z POWIĘKSZANIEM **/
/* small image */
span.autosize {
 position: relative;
 display: inline-block;
 max-width: 100%;
}
```

```
span.autosize img {
 max-width: 100%;
}

/* (hidden) zoom link */
span.autosize span {
 visibility: hidden;
 background-color: rgba(0,0,0,.25);
 text-align: center;
 display: block;
 width: 100%;
 left: 0;
}
span.autosize span a {
 font-weight : bold;
}
/*span.autosize span a:before {
 content: "POWIĘKSZ";
}*/

/* (hidden) zoom link - overlay version */
span.autosize span.overlay {
 display: none;
 position: absolute;
 width: 60%;
 left: 20%;
 height: 60%;
 top: 20%;
 vertical-align: middle;
}
span.autosize span.overlay a {
 position: relative;
 top: 45%;
 font-size : large;
}

/* hover - show zoom link */
span.autosize:hover span {
 visibility:visible;
 display:block;
}

/* target - show full image, hide zoom link */
span.autosize:target {
 max-width: 9999%;
}
span.autosize:target img {
 max-width: 9999%;
}
span.autosize:target span {
 visibility: hidden
}
```

LAYOUT_DYNAMIC.CSS

```
div.tresc {
 margin-left: 25px;
}
```

```
div.menu {
 position: fixed;
 top: 0;
 left: 0;
 width: 20px;
 height: 100%;

overflow: hidden;
 background: url("menu.png") repeat #000000;
}
div.menu * {
 visibility: hidden;
}
div.menu p.menu_info {
 color: white;
 font-weight: bold;
 display: block;
 visibility: visible;
 text-indent: 0;
 text-align: center ;
 background: #777777;
}


div.menu:hover {
 position: fixed;
 top: 0;
 left: 0;
 width: 180pt;
 padding: 5pt;
 height: 100%;

overflow: auto;
 background: #f0f8ff;
}
div.menu:hover * {
 visibility: visible;
}
div.menu:hover p.menu_info {
 display: none;
 visibility: hidden;
}

@media print {
 div.menu {
  display: none ;
  visibility: hidden ;
 }
}
```

GENERATE_GOOGLE_SITEMAP.SH

```
#!/bin/bash

echo '<?xml version="1.0" encoding="UTF-8"?>
<urlset
```

```
  xmlns="http://www.google.com/schemas/sitemap/0.84"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.google.com/schemas/sitemap/0.84
                      http://www.google.com/schemas/sitemap/0.84/sitemap.xsd">'

TZ=UTC find . -type f -a ! -path './archiwum*' -a ! -path './MIRRORS*' -a ! -path '*/.*' -a ! ⤸
-path './~rrp*' -exec ls -ld --time-style=+"%FT%TZ" \{\} \; |
grep -v 'page[.][A-Z-]+[.]xhtml' | cut -d' ' -f6- | awk ' {
  gsub("^./", "", $2)
  printf("<url><loc>http://www.opcode.eu.org/%s</loc>", $2)
  printf("<lastmod>%s</lastmod><priority>0.5000</priority></url>\n", $1)
 }'

echo '</urlset>'
```

SOFTDB.XSL

```
<?xml version="1.0"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fn="http://www.w3.org/2005/xpath-functions"
exclude-result-prefixes="fn"
>
 <xsl:output method='xml' encoding='utf-8' />
 <!-- zaleznie od trybu -> ustawiane przez opcje parsera <xsl:strip-space elements="*" /> -->

<!-- parametry globalne arkusza: -->
 <xsl:param name="cat_name" />



<!-- element główny -->
 <xsl:template match="soft_db">
  <!-- dla każdej kategori której nazwa zgadza się z podaną w parametrze -->
  <!-- ta petla ma zazwyczaj JEDEN obieg,
       ale jej zastosowanie jest wygodna metoda sprawdzania warunku na nazwe kategorii -->
  <xsl:for-each select="soft_cat[@name=$cat_name]">
   <!-- stosujemy odpowiedni szablon -->
   <xsl:apply-templates select="." />
  </xsl:for-each>
 </xsl:template>

<!-- przetwarzamy wszystkie elementy w kategorii, w tym podkategorie
      poprzez odwolania do odpowiednich szablonow -->
 <xsl:template match="soft_cat">
  <!-- umieszczamy opis kategorii ... -->
  <xsl:apply-templates select="desc" />
  <ul>
   <!-- ... i dla każdego elementu w kategori ... -->
   <xsl:for-each select="soft_cat|app_link|app">
    <!-- ... posortowanego w kolejniości alfabetycznej ... -->
    <!--xsl:sort select="@name" /-->
    <!-- ... stosuj odpowiedni szablon,
         umieszczając jego wynik w elemencie listy -->
    <li><xsl:apply-templates select="." /></li>
   </xsl:for-each>
  </ul>
 </xsl:template>
```

```
<!-- dla elementów app_link wyszukujemy w całej bazie elementu app o takiej samej nazwie -->
>
 <xsl:template match="app_link">
  <!-- pewnie dałoby sie zapisać warunek for-each bez tworzenia tej zmiennej,
       ale tak jest chyba czytelniej itd ... -->
  <xsl:variable name="name">
   <xsl:value-of select="@name"/>
  </xsl:variable>
  <!-- ta petla ma zazwyczaj JEDEN obieg,
       ale jej zastosowanie jest wygodna metoda znalezienia odpowiedniego tagu -->
  <xsl:for-each select="//app[@name=$name]">
   <xsl:apply-templates select="." />
  </xsl:for-each>
 </xsl:template>

<!-- dotarliśmy do elementu opisującego dany program, w tym miejscu generujemy
      zawartość <li/> opisującego dany program w oparciu o informacje z bazy ... -->
 <xsl:template match="app">
  <xsl:variable name="prio"><xsl:choose>
   <xsl:when test="@priority">
    <xsl:value-of select="@priority"/>
   </xsl:when>
   <xsl:otherwise>
    <xsl:text>normal</xsl:text>
   </xsl:otherwise>
  </xsl:choose></xsl:variable>
  <xsl:choose>
   <xsl:when test="@type='cmd'">
    <code class="{$prio}-app">
     <xsl:value-of select="@name" />
     <xsl:if test="@attr">
      <xsl:text> </xsl:text>
      <var>
       <xsl:value-of select="@attr" />
      </var>
     </xsl:if>
    </code>
   </xsl:when>
   <xsl:when test="homepage">
    <a class="ext_homepage {$prio}-app" href="{homepage/text()}">
     <xsl:value-of select="@name" />
    </a>
   </xsl:when>
   <xsl:otherwise>
    <span class="{$prio}-app">
     <xsl:value-of select="@name" />
    </span>
   </xsl:otherwise>
  </xsl:choose>

  <!-- podajemy w nawiasie dodatkow nazwy komend i programu -->
  <xsl:if test="alt_name|cmd_name">
   <xsl:text> (</xsl:text>
    <xsl:for-each select="cmd_name|alt_name">
     <xsl:choose>
      <xsl:when test="name() = 'cmd_name'">
       <code><xsl:apply-templates select="." /></code>
      </xsl:when>
      <xsl:otherwise>
```

```
        <xsl:apply-templates select="." />
      </xsl:otherwise>
    </xsl:choose>
    <xsl:if test="position() != last()">
      <xsl:text>, </xsl:text>
    </xsl:if>
  </xsl:for-each>
  <xsl:text>)</xsl:text>
</xsl:if>

<!-- opis -->
<xsl:text> - </xsl:text><xsl:apply-templates select="desc" />

<!-- id_link'i -->
<xsl:if test="id_link">
  <br /><xsl:text>Dodatkowe informacje: </xsl:text>
  <xsl:apply-templates select="id_link" mode="full_rewrite" />
</xsl:if>

<!-- dodatkowe strony domowe -->
<xsl:if test="alt_homepage or (homepage and @type='cmd')">
  <br /><xsl:text>Homepage: </xsl:text>
  <xsl:if test="@type='cmd'">
    <xsl:for-each select="homepage">
      <a class="ext_homepage" href="{text()}">
        <xsl:value-of select="text()" />
      </a>
    </xsl:for-each>
  </xsl:if>
  <xsl:for-each select="alt_homepage">
    <a class="ext_homepage" href="{text()}">
      <xsl:value-of select="text()" />
    </a>
  </xsl:for-each>
</xsl:if>

<!-- linki do paczek Debiana -->
<xsl:if test="deb">
  <br /><xsl:text>Debian: </xsl:text>
  <xsl:for-each select="deb">
    <xsl:choose>
      <xsl:when test="@repo">
        <xsl:value-of select="text()" />
        <xsl:text>@</xsl:text>
        <xsl:value-of select="@repo" />
      </xsl:when>
      <xsl:otherwise>
        <a class="ext_homepage" href="http://packages.debian.org/{text()}">
          <xsl:value-of select="text()" />
        </a>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:text> </xsl:text>
  </xsl:for-each>
</xsl:if>

<xsl:apply-templates select="id_link" mode="full_rewrite" />
</xsl:template>
```

```
<!-- tryb pełnego przepisywania
      na potrzeby wstawiania fragmentów XML z bazy danych do pliku wynikowego -->
 <xsl:template match="*|@*|processing-instruction()|text()" mode="full_rewrite">
  <xsl:copy>
   <xsl:apply-templates mode="full_rewrite" select="*|@*|processing-instruction()|text()" />
  </xsl:copy>
 </xsl:template>
 <!-- szablon dla tagów których zawartość ma być przepisywana bez zmian -->
 <xsl:template match="desc">
  <xsl:apply-templates select="*|text()" mode="full_rewrite" />
 </xsl:template>
</xsl:stylesheet>
```

BASE.XSL

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fn="http://www.w3.org/2005/xpath-functions"
exclude-result-prefixes="fn"
> <!-- tak naprawdę 2.0 (gdy używany tryb latex), ale xsltproc (używany dla trybu xhtml) ⤸
nie che przetwarzać 2.0 -->

<xsl:output method='xml' encoding='utf-8' />
 <!-- zależnie od trybu -> ustawiane przez opcje parsera <xsl:strip-space elements="*" /> -->

<!-- parametry globalne arkusza: -->
 <xsl:param name="tryb" /> <!-- tryb przetwarzania latex lub xhtml -->
 <xsl:param name="page_url" /> <!-- bazowy adres strony dla trybu latex -->
 <xsl:param name="photo_dir" /> <!-- adres katalogu z zdjęciami i galeriami względem ⤸
katalogu głównego strony (zaczynający się i kończący /) -->

<!-- definijemy zmienne które bedziemy używać jako nową linię i podwójną nową linię ⤸
-->
 <xsl:variable name="LF"><xsl:text>
</xsl:text></xsl:variable>
 <xsl:variable name="LF_2">
  <xsl:value-of select="$LF"/>
  <xsl:value-of select="$LF"/>
 </xsl:variable>

<!-- TYTUŁY / NAGŁÓWKI:
  <title> konwertowane na stosowny <headX>
 -->
 <xsl:template match="head0" mode="xhtml">
  <h1>
   <xsl:attribute name="class">center <xsl:copy-of select="@class" /></xsl:attribute>
   <xsl:copy-of select="attribute::node()" />
   <xsl:apply-templates mode="xhtml" select="node()"/>
  </h1>
 </xsl:template>
 <xsl:template match="head0" mode="latex">
  <xsl:text>\newpage\hypertarget{</xsl:text>
   <xsl:value-of select="@id" />
  <xsl:text>}{\part{</xsl:text>
   <xsl:apply-templates mode="latex" select="node()"/>
  <xsl:text>}}</xsl:text>
```

```
    <xsl:value-of select="$LF_2"/>
  </xsl:template>


<xsl:template match="head1" mode="xhtml">
    <h2>
      <xsl:copy-of select="attribute::node()" />
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </h2>
  </xsl:template>
  <xsl:template match="head1" mode="latex">
    <xsl:text>\hypertarget{</xsl:text>
      <xsl:value-of select="@id"/>
    <xsl:text>}{\chapter{</xsl:text>
      <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>}}</xsl:text>
    <xsl:value-of select="$LF_2"/>
  </xsl:template>

<xsl:template match="head2" mode="xhtml">
    <h3>
      <xsl:attribute name="class">center <xsl:copy-of select="@class" /></xsl:attribute>
      <xsl:copy-of select="attribute::node()" />
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </h3>
  </xsl:template>
  <xsl:template match="head2" mode="latex">
    <xsl:text>\hypertarget{</xsl:text>
      <xsl:value-of select="@id"/>
    <xsl:text>}{\section{</xsl:text>
      <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>}}</xsl:text>
    <xsl:value-of select="$LF_2"/>
  </xsl:template>

<xsl:template match="head3" mode="xhtml">
    <h4>
      <xsl:copy-of select="attribute::node()" />
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </h4>
  </xsl:template>
  <xsl:template match="head3" mode="latex">
    <xsl:text>\hypertarget{</xsl:text>
      <xsl:value-of select="@id"/>
    <xsl:text>}{\subsection{</xsl:text>
      <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>}}</xsl:text>
    <xsl:value-of select="$LF_2"/>
  </xsl:template>

<xsl:template match="head4" mode="xhtml">
    <h5>
      <xsl:copy-of select="attribute::node()" />
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </h5>
  </xsl:template>
  <xsl:template match="head4" mode="latex">
    <xsl:text>\hypertarget{</xsl:text>
      <xsl:value-of select="@id"/>
    <xsl:text>}{\subsubsection{</xsl:text>
```

```
    <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>}}</xsl:text>
   <xsl:value-of select="$LF_2"/>
  </xsl:template>

 <xsl:template match="head5" mode="xhtml">
   <h6>
    <xsl:copy-of select="attribute::node()" />
    <xsl:apply-templates mode="xhtml" select="node()"/>
   </h6>
  </xsl:template>
  <xsl:template match="head5" mode="latex">
   <xsl:text>\hypertarget{</xsl:text>
    <xsl:value-of select="@id"/>
   <xsl:text>}{\paragraph{</xsl:text>
    <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>}}</xsl:text>
   <xsl:value-of select="$LF_2"/>
  </xsl:template>

 <xsl:template match="head6" mode="latex">
   <xsl:text>\hypertarget{</xsl:text>
    <xsl:value-of select="@id"/>
   <xsl:text>}{\subparagraph{</xsl:text>
    <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>}}</xsl:text>
   <xsl:value-of select="$LF_2"/>
  </xsl:template>

 <xsl:template match="div[@id]" mode="latex">
   <xsl:text>\hypertarget{</xsl:text>
    <xsl:value-of select="@id"/>
   <xsl:text>}{</xsl:text>
    <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>}</xsl:text>
  </xsl:template>

 <xsl:template match="article" mode="xhtml">
   <xsl:apply-templates mode="xhtml" select="node()"/>
  </xsl:template>
  <xsl:template match="article" mode="latex">
   <xsl:apply-templates mode="latex" select="node()"/>
  </xsl:template>

 <!-- ODNOSNIKI:
   <wiki pl="link" en="link">text</wiki> - linki do Wikipedii
   <doc url="" mirror="">text</doc> - linki do materiałów archiwizowanych
   <datasheet id=""> - linki do dokumentacji układów elektronicznych (wyszukiwanie na ⤸
   EleNota po zawartości atrybutu id)
   <homepage> - linki do stron domowych projektów, oprogramowania itp postaci:
    <homepage url="adres" mirror="">opis</homepage> lub <homepage mirror="">adres</homepage>
   <id_link id="identyfikator:wewnetrzny"> - obsługiwany w skrypcie AWK
   <photo_link tag="tag">opis</photo_link> - link do galerii zdjęć związanych z danym ⤸
   tagiem
   <photo_link path="sciezka_wzgledem katalogu z zdjęciami">opis</photo_link> - link do ⤸
   zdjęcia
   <sch_link name="nazwa_bez_rozszerzenia">opis</sch_link> - link do pliku .sch z dodaniem ⤸
   linków do wersji PDF i PNG
    na potrzeby generacji latex do tagu jest także automatycznie wstawiany dir=""
   <pcb_link name="nazwa_bez_rozszerzenia">opis</pcb_link> - link do pliku .pcb z dodaniem ⤸
```

```
  linków do wersji PDF
   na potrzeby generacji latex do tagu jest także automatycznie wstawiany dir=""
  <a> - standardowe linki HTML
-->
 <xsl:template match="wiki" mode="xhtml">
  <xsl:variable name="base_lang"><xsl:choose>
   <xsl:when test="@pl">pl</xsl:when>
   <xsl:when test="@en">en</xsl:when>
  </xsl:choose></xsl:variable>
  <a class="Wikipedia" href="http://{$base_lang}.wikipedia.org/wiki/{@*[local-name()⤸
  =$base_lang]}">
   <xsl:apply-templates mode="xhtml" select="node()"/>
  </a>
 </xsl:template>
 <xsl:template match="wiki" mode="latex">
  <xsl:variable name="base_lang"><xsl:choose>
   <xsl:when test="@pl">pl</xsl:when>
   <xsl:when test="@en">en</xsl:when>
  </xsl:choose></xsl:variable>
  <xsl:call-template name="latex_link">
   <xsl:with-param name="url">
    <xsl:text>http://</xsl:text>
    <xsl:value-of select="$base_lang"/>
    <xsl:text>.wikipedia.org/wiki/</xsl:text>
    <xsl:value-of select="@*[local-name()=$base_lang]"/>
   </xsl:with-param>
   <xsl:with-param name="txt" select="node()"/>
  </xsl:call-template>
 </xsl:template>

<xsl:template match="doc[@mirror]|homepage[@mirror]" mode="xhtml">
  <span class="linkWithMirror">
   <xsl:apply-templates mode="xhtml-reallinks" select="."/>
   <a class="mirror" href="[[!URLPREFIX!]]MIRRORS/{@mirror}">kopia lokalna</a>
  </span>
 </xsl:template>
 <xsl:template match="doc[not(@mirror)]|homepage[not(@mirror)]" mode="xhtml">
  <xsl:apply-templates mode="xhtml-reallinks" select="."/>
 </xsl:template>

<xsl:template match="doc" mode="xhtml-reallinks">
  <a href="{@url}" class="ext_doc">
   <xsl:apply-templates mode="xhtml" select="node()"/>
  </a>
 </xsl:template>
 <xsl:template match="doc" mode="latex">
  <xsl:call-template name="latex_link">
   <xsl:with-param name="url" select="@url"/>
   <xsl:with-param name="txt" select="node()"/>
  </xsl:call-template>
 </xsl:template>

<xsl:template match="homepage" mode="xhtml-reallinks">
  <a class="ext_homepage">
   <xsl:attribute name="href"><xsl:choose>
    <xsl:when test="@url">
     <xsl:value-of select="@url"/>
    </xsl:when><xsl:otherwise>
     <xsl:value-of select="node()"/>
    </xsl:otherwise>
```

```
      </xsl:choose></xsl:attribute>
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </a>
  </xsl:template>
  <xsl:template match="homepage" mode="latex">
    <xsl:call-template name="latex_link">
      <xsl:with-param name="url"><xsl:choose>
        <xsl:when test="@url">
          <xsl:value-of select="@url"/>
        </xsl:when><xsl:otherwise>
          <xsl:value-of select="node()"/>
        </xsl:otherwise>
      </xsl:choose></xsl:with-param>
      <xsl:with-param name="txt" select="node()"/>
    </xsl:call-template>
  </xsl:template>

<xsl:template match="datasheet" mode="xhtml">
    <a class="ext_doc" href="http://www.elenota.pl/search.php?man=--wszyscy--&amp;ile=50&amp;⤸
    start=0&amp;szuk={@id}">
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </a>
  </xsl:template>
  <xsl:template match="datasheet" mode="latex">
    <xsl:call-template name="latex_link">
      <xsl:with-param name="url">
        <xsl:text>http://www.elenota.pl/search.php?man=--wszyscy--\&amp;ile=50\&amp;start=0\&amp;⤸
        szuk=</xsl:text>
        <xsl:value-of select="@id"/>
      </xsl:with-param>
      <xsl:with-param name="txt" select="node()"/>
    </xsl:call-template>
  </xsl:template>

<xsl:template match="photo_link[@path]" mode="xhtml">
    <a class="int" href="[[!URLPREFIX!]]{$photo_dir}{@path}">
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </a>
  </xsl:template>
  <xsl:template match="photo_link[@path]" mode="latex">
    <xsl:call-template name="latex_link">
      <xsl:with-param name="url">
        <xsl:value-of select="$page_url" />
        <xsl:value-of select="$photo_dir" />
        <xsl:value-of select="@path" />
      </xsl:with-param>
      <xsl:with-param name="txt" select="node()"/>
    </xsl:call-template>
  </xsl:template>

<xsl:template match="photo_link[@tag]" mode="xhtml">
    <a class="int" href="[[!URLPREFIX!]]{$photo_dir}/{@tag}.0.[[!URLPOSTFIX!]]">
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </a>
  </xsl:template>
  <xsl:template match="photo_link[@tag]" mode="latex">
    <xsl:call-template name="latex_link">
      <xsl:with-param name="url">
        <xsl:value-of select="$page_url" />
        <xsl:value-of select="$photo_dir" />
```

```
      <xsl:value-of select="@tag" />
      <xsl:text>.0.[[!URLPOSTFIX!]]</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="txt" select="node()"/>
  </xsl:call-template>
 </xsl:template>

<xsl:template match="sch_link" mode="xhtml">
  <a class="int" href="./{@name}.sch">
    <xsl:apply-templates mode="xhtml" select="node()"/>
  </a> (<a class="int" href="./{@name}.pdf">PDF</a>, <a class="int" href="./{@name}.png">PNG<⤶
  /a>)
</xsl:template>
 <xsl:template match="sch_link" mode="latex">
  <xsl:call-template name="latex_link">
    <xsl:with-param name="url">
      <xsl:value-of select="$page_url" />
      <xsl:value-of select="@dir" />
      <xsl:value-of select="@name" />
      <xsl:text>.sch</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="txt" select="node()"/>
  </xsl:call-template>
  <xsl:text>(PDF: \url{</xsl:text>
    <xsl:value-of select="$page_url" />
    <xsl:value-of select="@dir" />
    <xsl:value-of select="@name" />
  <xsl:text>.pdf})</xsl:text>
 </xsl:template>

<xsl:template match="pcb_link" mode="xhtml">
  <a class="int" href="./{@name}.pcb">
    <xsl:apply-templates mode="xhtml" select="node()"/>
  </a> (<a class="int" href="./{@name}.pcb.pdf">PDF</a>)
</xsl:template>
 <xsl:template match="pcb_link" mode="latex">
  <xsl:call-template name="latex_link">
    <xsl:with-param name="url">
      <xsl:value-of select="$page_url" />
      <xsl:value-of select="@dir" />
      <xsl:value-of select="@name" />
      <xsl:text>.pcb</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="txt" select="node()"/>
  </xsl:call-template>
  <xsl:text>(PDF: \url{</xsl:text>
    <xsl:value-of select="$page_url" />
    <xsl:value-of select="@dir" />
    <xsl:value-of select="@name" />
  <xsl:text>.pcb.pdf})</xsl:text>
 </xsl:template>

<xsl:template match="a[@href]" mode="latex">
  <xsl:variable name="adres"><xsl:choose>
    <!-- delokalizowanie linków (usunięcie poczatkowego ./ -->
    <xsl:when test="starts-with(@href,'./')">
      <xsl:value-of select="substring-after(@href,'./')"/>
    </xsl:when>
    <xsl:when test="starts-with(@href,'[[!URLPREFIX!]]/[[!URLPOSTFIX!]]')">
      <xsl:value-of select="substring-after(@href,'[[!URLPREFIX!]]/[[!URLPOSTFIX!]]')"/>
```

```
    </xsl:when>
    <!-- dodanie $PAGE_URL w linkach globalnych -->
    <xsl:when test="starts-with(@href,'/')">
     <xsl:value-of select="concat($page_url, @href)"/>
    </xsl:when>
    <xsl:otherwise>
     <xsl:value-of select="@href"/>
    </xsl:otherwise>
   </xsl:choose></xsl:variable>

   <xsl:choose>
    <!-- gdy href jest lokalny to używamy \hyperlink -->
    <xsl:when test="starts-with($adres,'#')">
     <xsl:text>\hyperlink{</xsl:text> <!-- zamiast \ref bądź \autoref -->
      <xsl:value-of select="substring-after(@href,'#')"/>
     <xsl:text>}{</xsl:text>
      <xsl:apply-templates mode="latex" select="node()"/>
     <xsl:text>}</xsl:text>
    </xsl:when>

    <!-- gdy href taki sam jak opis to używamy \url{} -->
    <xsl:when test="$adres = .">
     <xsl:text>\url{</xsl:text>
      <xsl:value-of select="$adres"/>
     <xsl:text>}</xsl:text>
    </xsl:when>

    <!-- w pozostałych wypadkach używamy \href
     a adres linku podajemy w przypisach  -->
    <xsl:otherwise>
     <xsl:call-template name="latex_link">
      <xsl:with-param name="url" select="$adres"/>
      <xsl:with-param name="txt" select="node()"/>
     </xsl:call-template>
    </xsl:otherwise>
   </xsl:choose>
  </xsl:template>



<!-- OBRAZKI:
   <img_from_src name="nazwa_bez.rozszerzenia" type="rozszerzenie_pliku_zrodlowego" alt="opis ⤸
   obrazka" /> - obrazki generowane dynamicznie ze źródeł
    na potrzeby generacji latex do tagu jest także automatycznie wstawiany dir=""
   <sch_show name="" /> - pokazanie pliku sch (tylko tryb latex)
   <img> - standardowe obrazki HTML
-->
  <xsl:template match="img_from_src" mode="xhtml">
   <span class="autosize" id="{generate-id()}"><span><a href="#{generate-id()}">POWIĘKSZ</a><⤸
   /span><a class="int" href="./{@name}.{@type}"><img src="./{@name}.png" alt="{@alt}" ⤸
   style="max-width:100%" /></a></span>
  </xsl:template>
  <xsl:template match="img_from_src" mode="latex">
   <xsl:text>\href{</xsl:text>
    <xsl:value-of select="$page_url"/>
    <xsl:value-of select="@dir"/>
    <xsl:value-of select="@name"/>
    <xsl:text>.</xsl:text>
    <xsl:value-of select="@type"/>
   <xsl:text>}{\adjincludegraphics[max size={\linewidth}{0.9\textheight}]{</xsl:text>
```

```
    <xsl:value-of select="@name"/>
   <xsl:text>}}</xsl:text>
  </xsl:template>
  <xsl:template match="img" mode="latex">
   <xsl:text>\adjincludegraphics[max size={\linewidth}{0.9\textheight}]{</xsl:text>
    <xsl:value-of select="fn:replace(@src, '[.][^.]*$', '')"/>
   <xsl:text>}</xsl:text>
  </xsl:template>

 <xsl:template match="sch_show" mode="xhtml">
  </xsl:template>
  <xsl:template match="sch_show" mode="latex">
   <xsl:text>\textbf{\texttt{</xsl:text>
    <xsl:call-template name="latex_escape"><xsl:with-param name="txt" select="@name"/></xsl:
    call-template>
   <xsl:text>.sch}:\\}</xsl:text>
   <xsl:text>\adjincludegraphics[max size={\linewidth}{0.9\textheight}]{</xsl:text>
    <xsl:value-of select="@name"/>
   <xsl:text>}</xsl:text>
   <xsl:value-of select="$LF_2"/>
  </xsl:template>

 <xsl:template match="span[@latex_img_width]" mode="xhtml">
   <xsl:apply-templates mode="xhtml" select="node()" />
  </xsl:template>
  <xsl:template match="span[@latex_img_width]" mode="latex">
   <xsl:if test="@class='left'">
    <xsl:text>\begin{wrapfigure}{l}{</xsl:text>
   </xsl:if>
   <xsl:if test="@class='right'">
    <xsl:text>\begin{wrapfigure}{r}{</xsl:text>
   </xsl:if>
   <xsl:value-of select="2.6 * @latex_img_width div 150"/>
   <xsl:text>cm}</xsl:text>

   <xsl:apply-templates mode="latex" select="node()" />

   <xsl:text>\end{wrapfigure}</xsl:text>
  </xsl:template>


 <!-- FORMATOWANIA SPECJALNE (1):
   <man> - strony manuali, opcjonalnie może mieć atrybut section wskazujący rozdział
   manuala
   <kbd class="path"> - ścieżki
   <kbd class="cmd_help"> - komendy zawierające dodatkowe info - źródła informacji typu
   info, czy -help
   <kbd class="path_help"> - ścieżki do dokumentacji

   <code> <pre> - kody skryptów, wpisy konfiguracyjne, komendy, ...
   <var> <samp> - zmienne i przykłady w ramach powyższych
 -->
  <xsl:template match="man" mode="xhtml">
   <kbd class="cmd_help">
    <xsl:text>man </xsl:text>
    <xsl:value-of select="@section"/>
    <xsl:text> </xsl:text>
    <xsl:apply-templates mode="xhtml" select="node()"/>
   </kbd>
  </xsl:template>
```

```
  <xsl:template match="man" mode="latex">
    <xsl:text>\mintinline{sh}{man </xsl:text>
      <xsl:value-of select="@section"/>
      <xsl:text> </xsl:text>
      <xsl:apply-templates mode="latex-special" select="text()|var|samp|span"/>
    <xsl:text>}</xsl:text>
  </xsl:template>
  <xsl:template match="kbd|code" mode="latex">
    <xsl:text>\mintinline{txt}{</xsl:text>
      <xsl:apply-templates mode="latex-special" select="text()|var|samp|span"/>
    <xsl:text>}</xsl:text>
  </xsl:template>
  <xsl:template match="pre" mode="latex">
    <xsl:text>\begin{minted}{txt}</xsl:text>
      <xsl:if test="not(starts-with(text()[1],$LF))">
        <xsl:value-of select="$LF"/>
      </xsl:if>
      <xsl:apply-templates mode="latex-special" select="text()|var|samp|span"/>
      <xsl:if test="not(ends-with(text()[1],$LF))">
        <xsl:value-of select="$LF"/>
      </xsl:if>
    <xsl:text>\end{minted}</xsl:text>
    <xsl:value-of select="$LF_2"/>
  </xsl:template>

<xsl:template match="var|samp" mode="latex">
    <xsl:text>\textit{</xsl:text>
      <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>}</xsl:text>
  </xsl:template>
  <!-- konieczne inne traktowanie dla span i text()
       gdyż w minted i mintinline wyłączona interpretacja specjalnych -->
  <xsl:template match="text()|var|samp|span" mode="latex-special">
    <xsl:value-of select="."/>
  </xsl:template>



<!-- RÓWNANIA:
    <eq> - proste równania matematyczne - xhtml (<sub> i <sup>)
    <eq_latex> - złożone równania matematyczne - latex
               (dla wyjścia xhtml przetwarzany przez parse_eq_latexuation)
-->
  <xsl:template match="eq" mode="xhtml">
    <i>
      <xsl:attribute name="class">equation <xsl:copy-of select="@class" /></xsl:attribute>
      <xsl:copy-of select="attribute::node()" />
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </i>
  </xsl:template>
  <xsl:template match="eq" mode="latex">
    <xsl:text>$</xsl:text>
      <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>$</xsl:text>
  </xsl:template>
  <xsl:template match="eq//sub" mode="latex">
    <xsl:text>_{</xsl:text>
      <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>}</xsl:text>
  </xsl:template>
```

```
 <xsl:template match="eq//sup" mode="latex">
  <xsl:text>^{</xsl:text>
   <xsl:apply-templates mode="latex" select="node()"/>
  <xsl:text>}</xsl:text>
 </xsl:template>
 <xsl:template match="eq_latex" mode="latex">
  <xsl:text>$</xsl:text>
   <xsl:value-of select="."/>
  <xsl:text>$</xsl:text>
 </xsl:template>

<!-- FORMATOWANIA SPECJALNE (2):
  <dfn> - nazwy
  <cite> - elementy cytowane

  <strong> - wyroznione
  <em> - lekkie wyroznienie

  <TODO> - plany rozwojowe itp
  <tmp> - informacje tymczasowe
-->
 <xsl:template match="TODO" mode="xhtml">
  <xsl:choose>
   <xsl:when test="@intro">
    <p id="todo" class="center"><strong>
     <xsl:value-of select="@intro"/>
    </strong></p>
    <ul>
     <xsl:apply-templates mode="xhtml" select="node()"/>
    </ul>
   </xsl:when>
   <xsl:otherwise>
    <p id="todo">
     <xsl:apply-templates mode="xhtml" select="node()"/>
    </p>
   </xsl:otherwise>
  </xsl:choose>
 </xsl:template>
 <xsl:template match="TODO" mode="latex">
  <xsl:value-of select="$LF_2"/>
  <xsl:choose>
   <xsl:when test="@intro">
    <xsl:value-of select="@intro"/>
    <xsl:text>\begin{itemize}</xsl:text>
     <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>\end{itemize}</xsl:text>
   </xsl:when>
   <xsl:otherwise>
    <xsl:text>\textit{</xsl:text>
     <xsl:apply-templates mode="latex" select="node()"/>
    <xsl:text>}</xsl:text>
   </xsl:otherwise>
  </xsl:choose>
  <xsl:value-of select="$LF_2"/>
 </xsl:template>

<xsl:template match="tmp" mode="xhtml">
  <i>
   <xsl:attribute name="class">tmp <xsl:copy-of select="@class" /></xsl:attribute>
   <xsl:copy-of select="attribute::node()" />
```

```
    <xsl:apply-templates mode="xhtml" select="node()"/>
   </i>
  </xsl:template>
  <xsl:template match="dfn|cite|em|tmp" mode="latex">
   <xsl:text>\textit{</xsl:text>
    <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>}</xsl:text>
  </xsl:template>
  <xsl:template match="strong" mode="latex">
   <xsl:text>\textbf{</xsl:text>
    <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>}</xsl:text>
  </xsl:template>



<!-- ZNACZNIKI BLOKOWE ITP
   <p> <br> <ul> <ol> <li> - standardowe znaczniki HTML dla akapitów, łamań linii, list ⤸
  numerowanych i wypunktowywanych
-->
  <xsl:template match="p" mode="latex">
   <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:value-of select="$LF_2"/>
  </xsl:template>
  <xsl:template match="br" mode="latex">
   <xsl:text>\\</xsl:text>
   <xsl:value-of select="$LF"/>
  </xsl:template>

<xsl:template match="ul" mode="latex">
   <xsl:text>\begin{itemize}</xsl:text>
   <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>\end{itemize}</xsl:text>
   <xsl:value-of select="$LF_2"/>
  </xsl:template>
  <xsl:template match="ol" mode="latex">
   <xsl:text>\begin{enumerate}</xsl:text>
   <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:text>\end{enumerate}</xsl:text>
   <xsl:value-of select="$LF_2"/>
  </xsl:template>
  <xsl:template match="li" mode="latex">
   <xsl:text>\item </xsl:text>
   <xsl:apply-templates mode="latex" select="node()"/>
   <xsl:value-of select="$LF"/>
  </xsl:template>



<!-- TABELE:
   <table> - standardowy znacznik HTML do tabel
   <col> - znacznik z informacją o kolumnach, używany tylko dla potrzeb generacji tabel ⤸
  latex'owych
   <tr> - standardowy znacznik HTML do wierszy, usunięcie atrybutów (zwłaszcza latex="")
   <td> - standardowy znacznik HTML do komórek, usunięcie znaczników z atrybutem skip (⤸
  pochłoniętych przez rowspan) w trybie HTML (używane dla LateX)
-->
  <xsl:template match="col" mode="xhtml">
  </xsl:template>
  <xsl:template match="td[@skip]" mode="xhtml">
```

```
  </xsl:template>
  <xsl:template match="tr" mode="xhtml">
    <tr>
      <xsl:apply-templates mode="xhtml" select="node()"/>
    </tr>
  </xsl:template>
  <xsl:template match="skiptd" mode="xhtml">
  </xsl:template>

<xsl:template match="table" mode="latex">
    <xsl:text>\begin{adjustbox}{max size={\linewidth}{0.9\textheight}}\begin{tabu}</xsl:text>
    <xsl:choose>
      <xsl:when test="@latex">
        <xsl:value-of select="@latex"/>
      </xsl:when>
      <xsl:when test="@latexscale">
        <xsl:text> to </xsl:text>
        <xsl:value-of select="@latexscale"/>
        <xsl:text>\linewidth </xsl:text>
      </xsl:when>
      <xsl:otherwise>
        <xsl:text> spread 2pt </xsl:text>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:text>{</xsl:text>
    <xsl:apply-templates mode="latex-columninfo" select="col"/>
    <xsl:text>}</xsl:text>
    <xsl:apply-templates mode="latex" select="tr"/>
    <xsl:text>\taburulecolor{white}</xsl:text>
    <xsl:apply-templates mode="latex-specialrow" select="col"/>
    <xsl:text>\end{tabu}\end{adjustbox}</xsl:text>
    <xsl:value-of select="$LF_2"/>
  </xsl:template>
  <xsl:template match="col" mode="latex-columninfo">
    <xsl:choose>
      <xsl:when test="@latex">
        <xsl:value-of select="@latex"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:choose>
          <xsl:when test="../@latexborder='1'">
            <xsl:text>|X[c,p]</xsl:text>
            <xsl:if test="position() = last()">
              <xsl:text>|</xsl:text>
            </xsl:if>
          </xsl:when><xsl:otherwise>
            <xsl:text>X[c,p]</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
  <xsl:template match="col" mode="latex-specialrow">
    <xsl:if test="position() != last()">
      <xsl:text>&amp;</xsl:text>
    </xsl:if>
  </xsl:template>

<xsl:template match="tr" mode="latex">
    <xsl:apply-templates mode="latex" select="td"/>
```

```
    <xsl:text> \\</xsl:text>
    <xsl:if test="@latex">
      <xsl:value-of select="@latex"/>
    </xsl:if>
    <xsl:value-of select="$LF"/>
  </xsl:template>
  <xsl:template match="tr/td" mode="latex">
    <xsl:if test="@colspan">
      <xsl:text>\multicolumn{</xsl:text>
      <xsl:value-of select="@colspan"/>
      <xsl:text>}{c</xsl:text>
      <xsl:if test="../../@latexborder='1'">
        <xsl:text>|</xsl:text>
      </xsl:if>
      <xsl:text>}{</xsl:text>
    </xsl:if>
    <xsl:if test="@rowspan">
      <xsl:text>\multirow{</xsl:text>
      <xsl:value-of select="@rowspan"/>
      <xsl:text>}{</xsl:text>
      <xsl:value-of select="@colspan"/>
      <xsl:text>\tabucolX}{\centering{}</xsl:text>
    </xsl:if>
    <xsl:if test="not(@skip)">
      <xsl:apply-templates mode="latex" select="node()"/>
    </xsl:if>
    <xsl:if test="@rowspan">
      <xsl:text>}</xsl:text>
    </xsl:if>
    <xsl:if test="@colspan">
      <xsl:text>}</xsl:text>
    </xsl:if>
    <!-- gdy jest to nie ostatni <td> w danym <tr> to wypisujemy & -->
    <xsl:if test="position() != last()">
      <xsl:text>&amp;</xsl:text>
    </xsl:if>
  </xsl:template>



<!-- TAGI BLOKUJACE W WYBRANYCH TRYBACH:
    <only_pdf> - zawartość wstawiana tylko przy generowaniu pdf
    <only_xhtml>  - zawartość wstawiana tylko przy generowaniu html
-->
  <xsl:template match="only_pdf" mode="xhtml">
  </xsl:template>
  <xsl:template match="only_pdf" mode="latex">
    <xsl:apply-templates mode="latex" select="node()"/>
  </xsl:template>

<xsl:template match="only_xhtml" mode="xhtml">
    <xsl:apply-templates mode="xhtml" select="node()"/>
  </xsl:template>
  <xsl:template match="only_xhtml" mode="latex">
  </xsl:template>



<!-- DZIAŁANIA SPECJALNE DLA LATEX -->
```

```
<xsl:template name="latex_link" >
  <xsl:param name="url" /><xsl:param name="txt" />
  <xsl:param name="foot_url" select="$url" />
  <xsl:text>\href{</xsl:text>
   <xsl:value-of select="$url"/>
  <xsl:text>}{</xsl:text>
   <xsl:apply-templates mode="latex" select="$txt"/>
  <xsl:text>}\footnote{</xsl:text>
   <xsl:call-template name="latex_escape"><xsl:with-param name="txt" select="$foot_url"/><
   /xsl:call-template>
  <xsl:text>}</xsl:text>
 </xsl:template>

<xsl:template name="latex_escape" >
  <xsl:param name="txt" />
  <!-- uwaga na temat interpretacji poniższego kodu - xsl wymaga dodatkowo:
   * podwojenia \ po obu stronach,
   * zabezpieczenia $ poprzez poprzedzenie pojedynczym \ po obu stronach,
   * zabezpieczenia { i } poprzez poprzedzenie pojedynczym \ po lewej stronie,
  -->
  <xsl:variable name="tekst1" select="fn:replace($txt, '\\', '\$\\backslash\$')" />
  <xsl:variable name="tekst2" select="fn:replace($tekst1, '%', '\\%')" />
  <xsl:variable name="tekst3" select="fn:replace($tekst2, '#', '\\#')" />
  <xsl:variable name="tekst4" select="fn:replace($tekst3, '_', '\\_')" />
  <xsl:variable name="tekst5" select="fn:replace($tekst4, '&amp;', '\\&amp;')" />
  <xsl:variable name="tekst7" select="fn:replace($tekst5, '\}', '\\}')" />
  <xsl:variable name="tekst8" select="fn:replace($tekst7, '\{', '\\{')" />
  <xsl:variable name="tekst9" select="fn:replace($tekst8, '\^', '\\^{}')" />
  <xsl:variable name="tekstA" select="fn:replace($tekst9, '\$', '\\\$')" />
  <xsl:variable name="tekst"  select="fn:replace($tekstA, '\\\$\\backslash\\\$', 
  '\$\\backslash\$')" />
  <!-- uwaga na temat interpretacji powyższego kodu - xsl wymaga dodatkowo:
   * podwojenia \ po obu stronach,
   * zabezpieczenia $ poprzez poprzedzenie pojedynczym \ po obu stronach,
   * zabezpieczenia { i } poprzez poprzedzenie pojedynczym \ po lewej stronie,
  -->
  <xsl:value-of select="$tekst"/>
 </xsl:template>

<!-- w przypadku LaTeX zawartość tagów latex przenosimy bez zmian
      w przypadku XHTML tagi te nie powinny wystąpić w dokumencie źródłowym -->
 <xsl:template match="latex" mode="latex">
  <xsl:value-of select="."/>
 </xsl:template>
 <xsl:template match="latex" mode="xhtml">
 </xsl:template>

<!-- zastępujemy tag space zwykła spacją
      (używany w trybie latex do zabezpieczenia zwykłej pomiędzy dwoma tagami) -->
 <xsl:template match="space" mode="latex">
  <xsl:text> </xsl:text>
 </xsl:template>

<!-- dla nodów tekstowych wykonujemy zabezpieczenia znaków specjalnych -->
 <xsl:template match="text()" mode="latex">
  <xsl:call-template name="latex_escape"><xsl:with-param name="txt" select="."/></xsl:call-
  template>
 </xsl:template>

<!-- a dla jawnie niezdefiniowanych elementów nie wyświetlamy nic -->
```

```
 <xsl:template match="*|@*|processing-instruction()" mode="latex">
 </xsl:template>



<!-- DZIAŁANIA SPECJALNE DLA XHTML -->

<!-- w przypadku XHTML wszystko pozostałe przepisujemy bez zmian
      pozwala to na nie definiować tutaj opisu tagów które nie mają być modyfikowane -->
 <xsl:template match="*|@*|processing-instruction()|text()" mode="xhtml">
  <xsl:copy>
   <xsl:apply-templates mode="xhtml" select="*|@*|processing-instruction()|text()" />
  </xsl:copy>
 </xsl:template>
 <!-- warto zauważyć że taka instrukcja w połączeniu z zastosowaniem jej
      do wszystkich dzieci głównego elementu np. poprzez
   <xsl:template match="doc"><doc><xsl:apply-templates mode="xhtml" /></doc></xsl:template>
      spowoduje wypisanie dokumentu XML z wykonaniem tylko jawnie podanych modyfikacji
      i ewentualną modyfikacją formatowania
-->



<!-- ELEMENT GLOWNY -->
 <xsl:template match="document">
  <xsl:variable name="tryb_low"
   select="translate($tryb, 'ABCDEFGHIJKLMNOPRSTWUYZQX', 'abcdefghijklmnoprstwuyzqx')" />
  <xsl:choose>
   <xsl:when test="$tryb_low='xhtml'">
    <xsl:apply-templates mode="xhtml" />
   </xsl:when>
   <xsl:when test="$tryb_low='latex'">
    <xsl:apply-templates mode="latex" />
   </xsl:when>
  </xsl:choose>
 </xsl:template>
</xsl:stylesheet>
```

Moje projekty

Zobacz także:

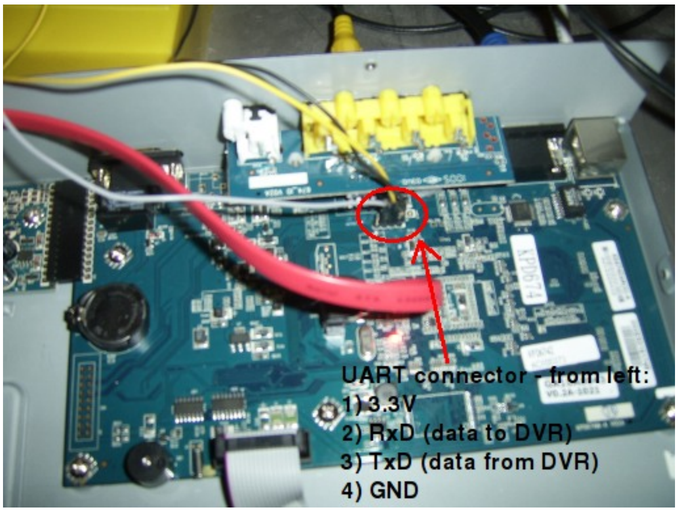
- ftdi_ds18b20.tar.gz[2159] - zestaw modułów dla Linuxa składających się na sterownik do obsługi termometru ds18b20 opartego na interfejsie OneWire z wykorzystaniem układu FTDI FT245BM do emulacji portu OneWire (rozpowszechniane na zasadach GNU GPL)
- Extension for MediaWiki - StringLoop[2160]
- Extension for MediaWiki - CategoryIndex[2161]

Rescue Avtech KPD674 after firmware upgrade fail

If firmware upgrade fail, but bootloader is OK you can tray USB flash on boot[2162]. Unfortunately, this method does not work sometimes (the screen displays a ”Reset Defaults …”, ”Complete Booting System …” and nothing about USB upgrade mode).

Serial console, u-boot menu and TFTP

Good news is: DVR have serial port and bootloader (u-boot) support it and TFTP. On my KPD674Z (KPD674AZ) serial with system console is located on 4-pins J5 connector (GND, RxD, TxD, +3.3V - see photo for detail). Serial voltage-level is 3.3V and baud-rate is 38400. If you are not connected VGA, then on early startup stage serial connection can be closed (`FATAL: term closed term_exitfunc: reset failed for dev UNKNOWN: Input/output error`) and you must reconnect to serial port. U-boot on serial console show:




```
012346
U-Boot 2008.10 (Dec 18 2009 - 10:07:53)
DRAM:  128 MB
Flash: 16 MB
Using default environment
flash no default environment
In:    serial
Out:   serial
Err:   serial
Net:   FTGMAC#0
checking keyboard is exist. iic type KEYBOARD_ID=00000060
boardid=0x02
force_output=0x30303030
Will set the following freq...
PLL1: 480 MHz, PLL3: 594 MHz, CPU freq: 480 MHz, AHB freq: 240 MHz, DDR freq: 480 MHz
go...
!VGA Detected(GPIO)
VGA Output
lcd200_ctrl_enable
Not BMP Type !!
Not support bmp 65535 Bits Per Pixel !!
Error bitmap format using default
bmp using RLE 8-bit/pixel
rle total length=76801
flash size=0x01000000
clock info:
PLL1: 480 MHz       PLL2: 750 MHz       PLL3: 594 MHz       DDR: 480 MHz
CPU : 480 MHz       HCLK: 240 MHz       PCLK: 120 MHz
UART: 118 MHz       IDE :  99 MHz       PCI :  42 MHz
MPEG: 198 MHz       H.264 enc: 198 MHz  H.264 dec: 198 MHz
Press SPACE to abort autoboot in 1 seconds
```

When you press space u-boot enter to boot menu:

```
Partition plan:
Name:BootLoader         offset:0x10000000 size:0x00040000  upgrade file:fa8181_bld.bin
Name:Initial bitmap     offset:0x10040000 size:0x00020000  upgrade file:custom_logo.bmp
Name:Factory default    offset:0x100A0000 size:0x00060000  upgrade file:
Name:Linux System       offset:0x10100000 size:0x00500000  upgrade file:kernel_16.bin
Name:Application        offset:0x10600000 size:0x00900000  upgrade file:AppImg_16.bin
Name:XML                offset:0x10F00000 size:0x00100000  upgrade file:xml_16.bin
Name:FULL               offset:0x10000000 size:0x01000000  upgrade file:full_16.bin
**********************************************
* AVC793 Please select option...
*   1 : Kermit Update BootLoader Firmware
```

```
*    2 : Kermit Update Kernel Firmware
*    3 : Kermit Update Application (AppImg)
*    4 : Kermit Update XML
*    5 : TFTP Configuration
*    6 : TFTP Update BootLoader Firmware
*    7 : TFTP Update Kernel Firmware
*    8 : TFTP Update Application (AppImg)
*    9 : TFTP Update XML
*    0 : TFTP Update Initialize Bitmap
*    a : TFTP Update FullInOne Image
*    b : Dump to MEM & Execute Use TFTP
*    c : Reboot
*    d : Start Linux
*    r : Reset Default
**********************************************
avtech>
```

Default TFTP configuration is: HSOT IP (DVR): 192.168.5.153 and SERVER IP: 192.168.5.152. So, install tftp on your PC (`aptitude install tftpd`) and configure network connection.

Hardware

First lines of kernel output allow hardware identification (it is GM8181[2163] system on chip for surveillance DVR based on ARM core with 128MB RAM and 5 serial interfaces; it's similar to the other grain-media ARMv5TE based DVRs SoC, especially gm8161):

```
Linux version 2.6.14 (root@debianaa) (gcc version 3.4.4) #367 PREEMPT Wed Jan 13 10:18:14
 ↪  CST 2010
CPU: FA626TEid(wb) [66056261] revision 1 (ARMv5TE)
Machine: Faraday GM8181
Memory policy: ECC disabled, Data cache writeback
GM8181 clock: PLL1OUT at 480 MHz, PLL2OUT at 750 MHz, PLL3OUT at 594 MHz
CPU0: D VIPT write-back cache
CPU0: I cache: 32768 bytes, associativity 4, 32 byte lines, 256 sets
CPU0: D cache: 32768 bytes, associativity 4, 32 byte lines, 256 sets
Built 1 zonelists
Kernel command line: mem=128M console=uart,shift,2,io,0xF9830000,38400 initcall_debug
 ↪  user_debug=31
PID hash table entries: 512 (order: 9, 8192 bytes)
GM Clock: CPU = 480 MHz, AHBCLK = 240 MHz
FTTMR010 timer 0 installed on IRQ 14, with clock 120000000 at 100 HZ.
IC: GM8181(C)
Console: colour dummy device 80x30
Early serial console at I/O port 0xf9830000 (options '38400', shift 2)
Dentry cache hash table entries: 16384 (order: 4, 65536 bytes)
Inode-cache hash table entries: 8192 (order: 3, 32768 bytes)
Memory: 128MB = 128MB total
Memory: 125056KB available (2567K code, 474K data, 1760K init)
Calibrating delay loop... 478.41 BogoMIPS (lpj=2392064)
Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
NET: Registered protocol family 16
GM AHB DMA Driver Init Ok
APB DMA Driver Init Ok
init APB DMA Interrupt mode
MM_DMA CH=0
GM PCI driver Init, version 2.0.0
GM PCI Clock = 66MHz
```

```
PCI: bus0: Fast back to back transfers disabled
Not Support Slot 0
SCSI subsystem initialized
NetWinder Floating Point Emulator V0.97 (double precision)
JFFS2 version 2.2. (NAND) (C) 2001-2003 Red Hat, Inc.
JFS: nTxBlock = 977, nTxLock = 7817
Serial: 8250/16550 driver $Revision: 1.3 $ 5 ports, IRQ sharing disabled
ttyS0 at I/O 0xf9830000 (irq = 9) is a 16550A
ttyS1 at I/O 0xf9840000 (irq = 10) is a 16550A
ttyS2 at I/O 0xf9850000 (irq = 20) is a 16550A
ttyS3 at I/O 0xf9880000 (irq = 21) is a 16550A
ttyS4 at I/O 0xf9890000 (irq = 22) is a 16550A
```

Firmware files


[2164]

Firmware package contains 4 files: fboot_4.bin, kernel_4.bin, AppImg_4.bin, xml_4.bin. Forget about "fboot_4.bin" - IF YOU HAVE WORKING U-BOOT DO NOT UPDATE THEM UNLESS IT IS ABSOLUTELY NECESSARY. All .bin files contains firmware header, for use by me kpd-674z_tylko_do_plyt_z.zip[2165] firmware version it length is 200 bytes. I found length of firmware header by comparing the kernel_4.bin and bootsImage from own kernel build. Probably length of header is write on first byte of .bin file. To extract bootable images you can use `dd if=kernel_4.bin of=/srv/tftp/kernel.img skip=200 bs=1`.

Avtech provided kernel source[2166]. On gm8161-sdk-v161[2167] are available compile tools (especially uClibc-toolchain.tgz) and other version of kernel source code for this platform. After installation (in /opt) uClibc-toolchain and set PATH (`PATH="/opt/toolchain/bin:$PATH"; export PATH`) you can build own kernel boot image: `make bootsImage` and additional modules (from gm8161-sdk) via `./build` script in module directory.

Extracted or own-made kernel image can be used to flash DVR with u-boot and TFTP (or can be boot directly from TFTP via u-boot):

```
avtech>7
Tftp upgrade Linux System
HSOT IP 192.168.5.153:
SERVER IP 192.168.5.152:
File name kernel_16.bin:
Using eth0 device
TFTP from server 192.168.5.152; our IP address is 192.168.5.153
Filename 'kernel_16.bin'.
Load address: 0x2000000
Loading: t RD_REQ, file: kernel_16.bin
#################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
```

```
         #################################################################
         ##################################
done
Bytes transferred = 3164300 (30488c hex)
upgrade(y/n):y
upgrade fw ...
upgrade Linux System
erase 0x10100000 +0x00500000
........................................ done
Erased 40 sectors
cp.b  0x02000000 0x10100000 $(filesize)
Copy to Flash... done
filesize=3164300
```

After extraction jffs2 filesystem image from .bin file (dd `if=AppImg_4.bin of=/srv/tftp/appimg.img skip=200 bs=1`) you can mount and modify starting script to debug and repair your DVR:

```
modprobe mtdblock
modprobe jffs2
modprobe mtdram total_size=7579 erase_size=256
dd if=/srv/tftp/appimg.img of=/dev/mtdblock0
mount -t jffs2 /dev/mtdblock0 /mnt/
mkdir /tmp/dvrapp
cp -a /mnt/ /tmp/dvrapp
cd /tmp/dvrapp
# dvrApp is main boot script in AppImg
# adding /bin/sh execution you can enter to shell on DVR ...
vi dvrApp
mkfs.jffs2 -o /srv/tftp/appimg.my1.img -r .
```

Original or modify filesystem image can be used to flash AppImg on DVR with u-boot and TFTP:

```
avtechi>8
Tftp upgrade Application
HSOT IP 192.168.5.153:
SERVER IP 192.168.5.152:
File name AppImg_16.bin:
Using eth0 device
TFTP from server 192.168.5.152; our IP address is 192.168.5.153
Filename 'AppImg_16.bin'.
Load address: 0x2000000
Loading: t RD_REQ, file: AppImg_16.bin
#################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
```

```
        ################################################################
        ################################################################
        ################################################################
        #####################
done
Bytes transferred = 7760696 (766b38 hex)
upgrade(y/n):y
upgrade fw ...
upgrade Application
erase 0x10600000 +0x00900000
.................................................................... done
Erased 72 sectors
cp.b  0x02000000 0x10600000 $(filesize)
Copy to Flash... done
filesize=7760696
```

In **xml_4.bin** isn't filesystem image, but is only one xml file, so it can't be flash directly by u-boot. Best way to update this file is copy them to DVR (for example via USB drive) and extract to /mnt/database/xml with DVRs tools:

```
mkdir /tmp/a
mount /dev/sdb /tmp/a  # mount USB
# extracting XML file from .bin file
/bin/upgradeHdr -s -i/tmp/a/xml_4.bin -o/tmp/config.xml -fxml -pH264DVRAV076_4 -vxml
# remove old config and import config from extracted file
rm -fr /mnt/database/xml/*
/mnt/mtd/Dvr/XmlFileM upload /tmp/config.xml
```

After restoring XML badabase you can check structure of /mnt/database/:

```
/ # ls -lR /mnt/database/
/mnt/database/:
drwx------    2 root     root            0 Oct 18 17:37 eventlog
drwx------    2 root     root            0 Oct 18 21:13 xml
/mnt/database/eventlog:
-rw-------    1 root     root          804 Oct 18 21:08 log
/mnt/database/xml:
-rw-------    1 root     root          272 Oct 18 21:13 General
-rw-------    1 root     root          215 Oct 18 21:13 LogLevels
-rw-------    1 root     root          440 Oct 18 21:13 Notification
-rw-------    1 root     root          117 Oct 18 21:13 MailLogd
-rw-------    1 root     root          675 Oct 18 21:13 Account
-rw-------    1 root     root           70 Oct 18 21:13 Bandwidth
-rw-------    1 root     root         4625 Oct 18 21:13 Network
-rw-------    1 root     root         1033 Oct 18 21:13 Properties
-rw-------    1 root     root         1271 Oct 18 21:13 Video
-rw-------    1 root     root          393 Oct 18 21:13 Brand
-rw-------    1 root     root           96 Oct 18 21:13 Audio
-rw-------    1 root     root        85060 Oct 18 21:13 DVR
-rw-------    1 root     root         8210 Oct 18 21:13 AVK
```

This method requires the modified application image, after install correct XML you can back to original AppImg or using own modification of AppImg. Using AppImg and XML from different version firmware packages is BAD idea.

Network streaming on KPD674Z

KPD674Z can stream video via:

— rtsp with h264: `rtsp://192.168.1.10/live/h264/ch1`
— mjpeg over http: `http://192.168.1.10/cgi-bin/guest/Video.cgi?media=MJPEG&channel=1&resolution=4CIF&sid=`
— jpeg over http: `http://192.168.1.10/cgi-bin/guest/Video.cgi?media=JPEG&channel=1&resolution=4CIF&sid=0.1`

More info about cgi usage in AV TECH Corporation CGI Command Set Specification[2168].

System obsługi DIRopu

System ten składający się z dwóch części:

- opartego na skryptach shellowych systemu przygotowania publikacji
- opartego na php systemu prezentacji publikacji

został opracowany na potrzeby projektu DIROP[2169] prowadzonego w ICM[2170] UW[2171].

Zamieszczam archiwum[2172] z skryptami php do prezentacji starodruków (wersja rozpakowana[2173]) oraz archiwum[2174] ze skryptami do przygotowania pozycji dodawanych do systemu prezentacji starodruków (wersja rozpakowana[2175]).

Dodatkowo uzupełnieniem systemu jest skrypt generujący informacje dla wystawienia OAI-PMH poprzez system wydany przez "Virginia Polytechnic Institute and State University" na licencji typu MIT/X11 dostępnych na http://www.dlib.vt.edu/projects/OAI/software/xmlfile/xmlfile.html[2176] - generate-oai-pmh-dc.php oraz drobny patch[2177] do tego systemu poprawiający drobny błąd w tym systemie (wychodzący przy testowaniu takiego repo)

layout.css

```
/***************************
 ** GLOWNY UKLAD DOKUMENTU **
 ***************************/

/** STRONA **/
body {
 padding: 0;
 margin: 1em;
 min-width: 81em; /* = div.left_menu:min-width / div.left_menu:width */

background-color: #eaead4;
 color: #333;
 font-family: Verdana, sans-serif;
}

/** NAGLOWKI **/
div.header {
 padding: 0;
 margin: 0;
}
div.system_message {
 padding: 0.5em;
 margin: 0.5em;
 border: 2px dotted red;
 text-align: center;
 color: #000000;
}

/** ZAWARTOSC STRONY (w tym menu) **/
div.page {
 position: relative;
 padding: 0;
 margin: 0;
}
div.left_menu {
 position: relative;
 padding: 0;
 margin: 0;
```

```
 float: left;
 width: 24.5%;
 min-width: 20em;
}
div.content {
 position: relative;
 padding: 0;
 margin: 0;
 float: right;
 width: 74.5%;
 min-width: 60em; /* = body:min-width * div.content:width */
}

/** STOPKA **/
div.footer {
 clear: both;
 padding: 0.5em;
 margin: 0;
 text-align: left;
}
```

TREE.CSS

```
/** LISTA NAWIGACJI **/
ul.navigation {
 list-style: none;
 padding-left: 1.2em;
 margin: 0;
}
ul.navigation ul {
 list-style: none;
 padding-left: 1.2em;
 margin: 0.1em;
}

/* wyglądy poszczególnych pozycji */
ul.navigation li.nav_entry_link {
 list-style: url(dot.gif);
}
/* dzięki stosowaniu ":before -> content" zamist "list-style" obrazki regują na onclic w JS ⤸
*/
ul.navigation li.nav_collapsed > span:before {
 margin-right: 0.5em;
 content: url(minus.gif);
}
ul.navigation li.nav_uncollapsed > span:before {
 margin-right: 0.5em;
 content: url(plus.gif);
}
ul.navigation li span {
 cursor: pointer;
}
ul.navigation li span:hover {
 background-color: #ffffff;
}
/* aby nie było problemów z wyłączonym JS robimy to w skrypcie
ul.navigation li.nav_collapsed ul {
 display: none;
```

```
}*/
span#nav_conttent_current {
 font-weight: bold;
}

p#navtree_func {
 text-align: right;
}
p#navtree_func span {
 /*color: #06c;*/
cursor: pointer;
}
p#navtree_func span:hover {
 background-color: #ffffff;
}
```

TREE.JS

```
/** tablica asocjacyjna: klucz = id pozycji spisu tresci,
    wartość == true => pozycja rozwinięta
    wartość == false => pozycja zwinięta
    brak wpisu w tablicy pozycja niezmieniona **/
var navtree_co_rozwiniete = new Object();

/** podstawowe funkcje operujące drzewkiem **/
function navtree_zwin(co) {
 if (co.className.match(/(^|\s)nav_uncollapsed(\s|$)/)) {
  co.className = "nav_collapsed";
  co.getElementsByTagName("ul")[0].style.display = "none";

  var id = parseInt(co.getAttribute("id").substr(9));
  navtree_co_rozwiniete[id] = false;
  return true;
 }
 return false;
}
function navtree_rozwin(co) {
 if (co.className.match(/(^|\s)nav_collapsed(\s|$)/)) {
  co.className = "nav_uncollapsed";
  co.getElementsByTagName("ul")[0].style.display = "list-item";

  var id = parseInt(co.getAttribute("id").substr(9));
  navtree_co_rozwiniete[id] = true;
  return true;
 }
 return false;
}
function navtree_save() {
 var napis = "navtree_list="
for (var klucz in navtree_co_rozwiniete) {
  if (navtree_co_rozwiniete[klucz])
   napis += klucz + "-t "
  else
   napis += klucz + "-f "
}
 document.cookie=napis;
}
```

```
/** funkcje skojarzone z zdarzeniami **/
function navtree_przelacz() {
 // zdarzenia sa do span, a my chemy operowac na li bedacym jego rodzicem
 navtree_rozwin(this.parentNode) || navtree_zwin(this.parentNode);
 navtree_save();
}
function navtree_collapse_all() {
 var spis_tresci_entry = document.getElementsByTagName("li");
 for(i=0; i < spis_tresci_entry.length; i++)
  navtree_zwin(spis_tresci_entry[i]);
 navtree_save();
}
function navtree_uncollapse_all() {
 var spis_tresci_entry = document.getElementsByTagName("li");
 for(i=0; i < spis_tresci_entry.length; i++)
  navtree_rozwin(spis_tresci_entry[i]);
 navtree_save();
}



/** funkcja inicjalizująca - dokonuje ustawienia skojarzeń dla zdarzeń
oraz zwija listę i ustawia zapamiętane wartości rozwinięć **/
function navtree_inicjalizuj() {
 var spis_tresci_entry = document.getElementsByTagName("li");
 for(i=0; i < spis_tresci_entry.length; i++) {
  if (spis_tresci_entry[i].className.match(/(^|\s)nav_collapsed(\s|$)/)) {
   // zwijamy pozycję (dzieki umieszczeniu tego tutaj a nie w arkuszu styli
   // eliminujemy problem gdy css właczone a js wyłaczona)
   spis_tresci_entry[i].getElementsByTagName("ul")[0].style.display = "none";
   // przypisujemy funkcje rozwijającą (do span - bo li sie zagniezdzaja)
   spis_tresci_entry[i].getElementsByTagName("span")[0].onclick = navtree_przelacz;
  } else if (spis_tresci_entry[i].className.match(/(^|\s)nav_uncollapsed(\s|$)/)) {
   // przypisujemy funkcję rozwijającą (do span - bo li sie zagniezdzaja)
   spis_tresci_entry[i].getElementsByTagName("span")[0].onclick = navtree_przelacz;
  }
 }
 var element = document.getElementById('navtree_coll');
 if (element != null) element.onclick = navtree_collapse_all;
 element = document.getElementById('navtree_uncoll');
 if (element != null) element.onclick = navtree_uncollapse_all;

// rozwinięcie bierzącej
 element = document.getElementById('nav_conttent_current');
 if (element != null) {
  var co = element.parentNode;
  while (co != null) {
   // nie chemy pamiętać tych rozwinieć dlatego nie navtree_rozwin(co) a ten if...
   if (co.className.match(/(^|\s)nav_collapsed(\s|$)/)) {
    co.className = "nav_uncollapsed";
    co.getElementsByTagName("ul")[0].style.display = "list-item";
   }
   co = co.parentNode;
  }
 }

// zapamietane z poprzedniego
 var ciasteczko = document.cookie.substr(document.cookie.search("navtree_list=")+13);
 ciasteczko = ciasteczko.substr(0, document.cookie.indexOf(';')-1);
```

```
 var ciasteczka = ciasteczko.split(" ");
 for(i=0; i < ciasteczka.length; i++) {
  var tmp = ciasteczka[i].split("-");
  element = document.getElementById('conttent_' + tmp[0]);
  if (element !=null && tmp.length>1)
   if (tmp[1].charAt(0)=="t") { // rozwiń
    navtree_rozwin(element);
   } else { // zwiń
    navtree_zwin(element);
   }
 }
}
```

GENERATE-OAI-PMH-DC.PHP

```
<?php

/** KONFIG **/
// łączenie się z bazą danych
define('CONNECTION_STRING', 'host=localhost port=5432 dbname=dirop user=diropusr');
// URL tego pliku - poprzedza ściezki wykozystywane w systemie
define('DIROP_SCRIPT_PATH', '/pl');
// URL serwera na którym zainstalowany jest system
define('DIROP_FULL_URL_PREFIX', 'http://dir.icm.edu.pl');



if ($_SERVER["argc"] == 2 ) {
 $volume_id = $_SERVER["argv"][1];
} else {
 echo "USAGE: php " . $_SERVER["argv"][0] . " VOLUME\n";
 exit;
}

define('BOOK_INFO_SELECT', 'b.book_id, b.single_vid, v.volume_id, b.title AS "b.title", b.url ⤦
AS "b.url", v.title AS "v.title", v.url AS "v.url", dc_title, dc_creator, ⤦
dc_subject_and_keywords, dc_description, dc_publisher, dc_contributor, dc_date, info_category,⤦
b.extra AS "b.extra"');
define('BOOK_INFO_SELECT_FROM', 'book AS b JOIN volume AS v ON v.book_id=b.book_id');

$baza_danych = pg_connect(CONNECTION_STRING) or error('db_connect');

$SQL = 'SELECT '. BOOK_INFO_SELECT .' FROM '. BOOK_INFO_SELECT_FROM ." WHERE v.⤦
volume_id=$volume_id";
$result = pg_query($baza_danych, $SQL) or error('db_query', $SQL);

$num_rows = pg_num_rows($result);
if ($num_rows == 1) {
 $BOOK_INFO = pg_fetch_array($result, null, PGSQL_ASSOC);

/// ustawienie zmiennych zależnych od wielovolumowości
 if ($BOOK_INFO['single_vid'] >= -1) {
  $name = $BOOK_INFO['v.url'] . ".xml";
  $url = DIROP_FULL_URL_PREFIX . DIROP_SCRIPT_PATH . "/" . $BOOK_INFO['v.url'];
 } else {
  $name = $BOOK_INFO['b.url'] . "-" . $BOOK_INFO['v.url'] . ".xml";
  $url = DIROP_FULL_URL_PREFIX . DIROP_SCRIPT_PATH . "/" . $BOOK_INFO['b.url'] . "/" . ⤦
  $BOOK_INFO['v.url'];
```

```
 }

$xml = '<?xml version="1.0"?>
<oaidc:dc xmlns="http://purl.org/dc/elements/1.1/"
          xmlns:oaidc="http://www.openarchives.org/OAI/2.0/oai_dc/"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
                              http://www.openarchives.org/OAI/2.0/oai_dc.xsd"
>
 <title xml:lang="pl">' . $BOOK_INFO['dc_title'] . '</title>
 <creator xml:lang="pl">' . $BOOK_INFO['dc_creator'] . '</creator>
 <subject xml:lang="pl">' . $BOOK_INFO['dc_subject_and_keywords'] . '</subject>
 <description xml:lang="pl">' . $BOOK_INFO['dc_description'] . '</description>
 <publisher xml:lang="pl">' . $BOOK_INFO['dc_publisher'] . '</publisher>
 <contributor xml:lang="pl">' . $BOOK_INFO['dc_contributor'] . '</contributor>
 <date xml:lang="pl">' . $BOOK_INFO['dc_date'] . '</date>
 <identifier>' . $url . '</identifier>
</oaidc:dc>
';
 $xml = str_replace ("&", "&amp;", $xml);
 $fp = fopen($name, 'w');
 fwrite($fp, $xml);
 fclose($fp);
 echo "OK: results for volume_id=$volume_id saved in $name \n";
} else {
 echo "ERROR: for volume_id=$volume_id pg_num_rows is $num_rows \n";
}

?>
```

Shell login for phpMyAdmin

Jest to prosty skrypt umożliwiający logowanie się do systemu phpMyAdmin z wykorzystaniem loginu i hasła do konta shelowego na którym jest plik .my.cnf zawierający dane niezbędne do uzyskania dostępu do MySQL. Plik należy umieścić tak aby był wykonywany jako skrypt CGI (i aby użytkownicy z niego korzystali ...), w skrypcie należy ustawić URL do systemu phpMyAdmin.

Uwaga: Jest to nieoficjalny patch mojego autorstwa. Autor nie ponosi żadnej odpowiedzialności za szkody powstałe w skutek wykorzystania zawartych tu poprawek.

POBIERZ[2178]

INDEX.PHP

```
#!/usr/bin/php-cgi
<?php header('Content-Type: text/html; charset=utf-8', 1);
/** Skrypt umożliwia logowanie się do systemu phpmyadmin uzywając hasła i loginu konta ⤸
shelowego
 *    na którym umieszczony jest stosowny plik .my.cnf zawierający nazwę uzytkownika i ⤸
 hasło MySQL
 *
 * UWAGA 1:
 *    wymagane ustawienia dla .my.cnf:
 *    plik powinien należeć do grupy na której pracuje ten skrypt
 *    (nie powinna być to grupa na której pracuje serwer WWW)
 *    oraz grupa ta powinna mieć prawo odczytu tego pliku */
// chmod 440 /home/xxx/.my.cnf; chgrp www-data /home/xxx/.my.cnf;
```

```
/** UWAGA 2:
 *   użytkownik na prawch którego pracuje skrypt powinien mieć prawo czytania /etc/shadow ⤸
 */
?>
<html>
 <head>
  <title>phpMyAdmin - logowanie PAM</title>
 </head>
 <body>
<?php
$SERWER_URL="/";
 $username = $_POST['username'];
 $password = $_POST['password'];

if ($username != "") {
  if (! pam_auth($username, $password, &$error)) {
   echo $error;
   exit;
  }

  //$db_user = shell_exec ("id -u $username");
  $db_user = shell_exec ("awk 'BEGIN {FS=\"=\"} /^user=/ {printf \$2}' /home/$username/.my.⤸
  cnf");
  $db_pass = shell_exec ("awk 'BEGIN {FS=\"=\"} /^password=/ {printf \$2}' /home/$username/.⤸
  my.cnf");

  echo "
  <p>Witaj $username. Zostaniesz zalogowany do systemu zarządzania bazami danych MySQL,
   na podstawie danych z pliku <code>~/.my.cnf</code> jako ${db_user}.</p>

  <p>Po zalogowaniu się możesz utworzyć bazę danych kilkając na link
   <font color=\"blue\">Bazy danych</font>, następnie wpisując w pole \"Utwórz nową bazę ⤸
   danych\"
   <b>${db_user}-XXX</b>, gdzie XXX jest twoją nazwą bazy danych i naciskając \"Utwórz\"<⤸
   /p>

  <form method=\"post\" action=\"${SERWER_URL}index.php\" name=\"login_form\" target=\"_top\">
   <input type=\"hidden\" name=\"pma_username\" value=\"$db_user\" />
   <input type=\"hidden\" name=\"pma_password\" value=\"$db_pass\" />
   <input type=\"hidden\" name=\"server\" value=\"1\" />
   <input type=\"hidden\" name=\"lang\" value=\"pl-utf-8\" />
   <input type=\"hidden\" name=\"convcharset\" value=\"iso-8859-1\" />
   <center><input type=\"submit\" value=\"Login\" id=\"buttonYes\" /></center>
  </form>";
 } else {
  echo "
  <p>W poniższe pola należy wpisać swój login i hasło do konta shelowego
   - system pobierze odpowiednie informacje o koncie MySQL.</p>
  <center>
  <form method=\"post\" action=\"\" name=\"login\" target=\"_top\">
  <table cellpadding=\"3\" cellspacing=\"0\">
   <tr>
    <td align=\"right\" bgcolor=\"#E5E5E5\"><b>Użytkownik: </b></td>
    <td align=\"left\" bgcolor=\"#E5E5E5\">
     <input type=\"text\" name=\"username\" size=\"24\" class=\"textfield\" />
    </td>
   </tr>
   <tr>
    <td align=\"right\" bgcolor=\"#E5E5E5\"><b>Hasło: </b></td>
    <td align=\"left\" bgcolor=\"#E5E5E5\">
```

```
      <input type=\"password\" name=\"password\" size=\"24\" class=\"textfield\" />
     </td>
    </tr>
    <tr>
     <td colspan=\"2\" align=\"center\">
      <input type=\"submit\" value=\"Login\" id=\"buttonYes\" />
     </td>
    </tr>
   </table></form></center>
   <p>Aby zalogować się bezpośrednio jako użytkownik MySQL należy przejść do
    <a href=\"${SERWER_URL}index.php\">tej</a> strony.</p>";
 }
?>
 </body>
</html>
```

GAME_SERWER_INFO.PHP

```
<?php
/**
 * skrypt służy do umieszczania wpisów informacyjnych w bazie danych
 * oraz wyświetlania zaakceptowanych wpisów, obsługuje także akceptację wpisów
 *
 * zawsze wyświetlany jest minimum 1 wpis (o najbliższej dacie),
 * wyświetlane są wszystkie w pisy o dacie w przeciągu najbliższych 24 h
 * wpisy o dacie o 1 godzinie w przeszłość są automatycznie kasowane
 *
 * system opracowany dla portalu http://emergency.gry-online.pl
 *
**/
?>

<html>
<head><title>system informacji o serwerach gry</title></head>
<body>

<?php

/// protrect for sql injection
function protect_sql($sql) {
 return strtr($sql, ";'\"()", "_____");
}


/** konfiguracja bazy danych **/

$db_host = "localhost";
$db_database="";
$db_user="";
$db_pass="";

$URL_SKRYPTU="";


/** INSTALACJA BAZY DANYCH

CREATE TABLE `serwery` (
```

```
 `SID`   INT NOT NULL AUTO_INCREMENT,
 `name` VARCHAR( 55 ) NOT NULL ,
 `nick` VARCHAR( 55 ) NOT NULL ,
 `mods` VARCHAR( 255 ) ,
 `time` INT NOT NULL ,
 `accept` INT(1) NOT NULL,

 PRIMARY KEY ( `DID` )
);

**/

function err_info($sql) {
    return "<p>" . mysql_error() . "<br />Błąd spowodowany poleceniem SQL:<br />" . $sql . 
    "</p>";
}

require_once("./inc/functions.inc.php");
function is_authorised() {
 //return check_login($_SESSION['username'], $_SESSION['md5_password']);
 return check_login($_COOKIE["username"], $_COOKIE["md5_password"]);
}

///
/// dodawanie nowego wpisu
///
if ($_POST["action"]=="add") {
 mysql_connect($db_host, $db_user, $db_pass);
 mysql_select_db($db_database);

$time = $_POST["date"] + $_POST["hour"] * 3600 + $_POST["minuts"] * 60;

$sql = "INSERT INTO serwery SET name='" . protect_sql($_POST["name"]) . "' , nick='" . 
 protect_sql($_POST["nick"]) . "' , mods='" . protect_sql($_POST["mods"]) . "' , time='" . 
 $time . "', accept='0'";
 $query = mysql_db_query($db_database, $sql) or die(err_info($sql));
 echo "<p>Informacja została pomyślnie dodana do bazy. Po zatwierdzeniu jej przez zespół 
 redakcyjny ukaże się na stronie.<br /><br /><a href=\"$URL_SKRYPTU\">powrót do strony 
 głównej</a></p>";

///
/// formularz dodawania
///
} else if ($_GET["action"]=="add_form") {
 ?>
 <p>Wpisz informacje o stawianym serwerze:</p>
 <form method="post" action="">
  <input type="hidden" name="action" value="add" />
  <p>Nazwa serwera: <input type="text" name="name" size="25" maxlength="25" />
   <br />Nick założyciela: <input type="text" name="nick" size="25" maxlength="25" />

   <br />Wykorzystywane mody: <input type="text" name="mods" size="25" maxlength="250" />

   <br />Data: <select name="date"> <?php
   $czas = time();
   $dzien = mktime (0, 0, 0, date('m', $czas), date('d', $czas), date('Y', $czas));
   for ($i=0; $i<14; $i++) {
    $czas_new = $dzien + 86400 * $i;
    echo "    <option value=\"$czas_new\">" . date('Y-m-d', $czas_new) . "</option>\n";
   }
```

```
   ?></select>
   <br />Godzina: <select name="hour">
    <option value="0">0</option>
    <option value="1">01</option>
    <option value="2">02</option>
    <option value="3">03</option>
    <option value="4">04</option>
    <option value="5">05</option>
    <option value="6">06</option>
    <option value="7">07</option>
    <option value="8">08</option>
    <option value="9">09</option>
    <option value="10">10</option>
    <option value="11">11</option>
    <option value="12">12</option>
    <option value="13">13</option>
    <option value="14">14</option>
    <option value="15">15</option>
    <option value="16">16</option>
    <option value="17">17</option>
    <option value="18">18</option>
    <option value="19">19</option>
    <option value="20">20</option>
    <option value="21">21</option>
    <option value="22">22</option>
    <option value="23">23</option>
   </select> Minuty: <select name="minuts">
    <option value="0">00</option>
    <option value="5">05</option>
    <option value="10">10</option>
    <option value="15">15</option>
    <option value="20">20</option>
    <option value="25">25</option>
    <option value="30">30</option>
    <option value="35">35</option>
    <option value="40">40</option>
    <option value="45">45</option>
    <option value="50">50</option>
    <option value="55">55</option>
    <option value="60">60</option>
   </select>

   <br /><br /><input type="submit" value="Dodaj" />
  </p>
 </form>
 <?php

///
/// lista wpisów do moderacji
///
} else if ($_GET["action"]=="show_moderate" && is_authorised()) {
 mysql_connect($db_host, $db_user, $db_pass);
 mysql_select_db($db_database);

echo "<h2>Wpisy czekające na zaakceptowanie</h2>";
 $sql = "SELECT * FROM serwery WHERE accept='0'";
 $query = mysql_db_query($db_database, $sql) or die(err_info($sql));
 while ($serwer = mysql_fetch_array($query)) {
  echo "<p>Nazwa: " . $serwer["name"] . "<br>Nick: " . $serwer["nick"];
  echo "<br>Mody: " . $serwer["mods"] . "<br>Data: " . date('Y-m-d H:i', $serwer["time"]);;
```

```
  echo "<a href=\"?action=accept&amp;SID=" . $serwer["SID"] . "\">Zaakceptuj</a>";
  echo "<a href=\"?action=delete&amp;SID=" . $serwer["SID"] . "\">Usuń</a>";
 }

echo "<h2>Wpisy zaakceptowane</h2>";
 $sql = "SELECT * FROM serwery WHERE accept='1'";
 $query = mysql_db_query($db_database, $sql) or die(err_info($sql));
 while ($serwer = mysql_fetch_array($query)) {
  echo "<p>Nazwa: " . $serwer["name"] . "<br>Nick: " . $serwer["nick"];
  echo "<br>Mody: " . $serwer["mods"] . "<br>Data: " . date('Y-m-d H:i', $serwer["time"]);;
  echo "<a href=\"?action=delete&amp;SID=" . $serwer["SID"] . "\">Usuń</a>";
 }


///
/// akceptacja wpisu
///
} else if ($_GET["action"]=="accept" && is_authorised()) {
 mysql_connect($db_host, $db_user, $db_pass);
 mysql_select_db($db_database);

$sql = "UPDATE serwery SET accept='1' WHERE SID='" . $_GET["SID"] . "'";
 $query = mysql_db_query($db_database, $sql) or die(err_info($sql));
 echo "<p>Zaakceptowany<br /><br /><a href=\"?action=show_moderate\">powrót do listy</a></p>⤸
 ";

///
/// potwierdzenie usuniecia wpisu
///
} else if ($_GET["action"]=="delete" && is_authorised()) {
 echo "<h2>Na pewno usunąć?</h2><p><a href=\"?action=delete_yes&amp;SID=" . $_GET["SID"] . ⤸
 "\">TAK</a><br /><br /><a href=\"?action=show_moderate\">NIE</a></p>";

///
/// usuniecie wpisu
///
} else if ($_GET["action"]=="delete_yes" && is_authorised()) {
 mysql_connect($db_host, $db_user, $db_pass);
 mysql_select_db($db_database);

$sql = "DELETE FROM serwery WHERE SID='" . $_GET["SID"] . "'";
 $query = mysql_db_query($db_database, $sql) or die(err_info($sql));
 echo "<p>Usunięty<br /><br /><a href=\"?action=show_moderate\">powrót do listy</a></p>";


///
/// wyświetlane wpisy
///
} else {
 mysql_connect($db_host, $db_user, $db_pass);
 mysql_select_db($db_database);

$czas_del = time() - 3600;
 $czas_show = time() + 86400;
 $time_show2 = time() + 1209600;
 $i=0; $ctrl=0;

$sql = "SELECT * FROM serwery WHERE accept='1' ORDER BY time DESC";
 $query = mysql_db_query($db_database, $sql) or die(err_info($sql));
 while ($serwer = mysql_fetch_array($query)) {
```

```
    $i++;
    /// jeżeli założony godzinę temu usuwamy
    if ($serwer["time"] < $czas_del) {
     $sql2 = "DELETE FROM serwery WHERE SID='" . $serwer["SID"] . "'";
     $query2 = mysql_db_query($db_database, $sql2) or die(err_info($sql2));

    /// Pokazujemy najbliższy lub wszystkie z najbliższych 24 godzin
    } else if ( $serwer["time"] < $czas_show || ($i == 1 && $serwer["time"] < $time_show2 ) {
     echo "<p>Nazwa: " . $serwer["name"] . "<br>Nick: " . $serwer["nick"];
     echo "<br>Mody: " . $serwer["mods"] . "<br>Data: " . date('Y-m-d H:i', $serwer["time"]);
     $ctrl=1;
    }
   }
  if ($ctrl==0) {
   echo "<p>brak serwerow w przeciagu najblizszych 1209600 sekund ;-)</p>"
 }
 }

?>
</body>
</html>
```

NEWS.PHP

```
<?php

/**
 * Prosty system news'owy wykorzystujący bazę danych MySQL
**/

$admin_pass=""; // haslo umozliwiajace edycje niezaleznie od autora

// informacje konfiguracyjne dostepu do bazy danych
$db_host = "localhost";
$db_database="www-news";
$db_user="";
$db_pass="";

/* INSTALACJA BAZY DANYCH

CREATE TABLE `news` (
 `NID` INT NOT NULL AUTO_INCREMENT,
 `title` VARCHAR( 255 ) NOT NULL ,
 `UID` INT NOT NULL ,
 `time` INT NOT NULL ,
 `body` TEXT NOT NULL ,
 PRIMARY KEY ( `NID` )
);

CREATE TABLE `users` (
 `UID` INT NOT NULL AUTO_INCREMENT,
 `username` VARCHAR( 255 ) NOT NULL ,
 `password` VARCHAR( 255 ) NOT NULL ,
 `status` VARCHAR( 255 ) NOT NULL ,
 `obrazek` VARCHAR( 255 ) NOT NULL ,
 `email` VARCHAR( 255 ) NOT NULL ,
 `jabber` VARCHAR( 255 ) NOT NULL ,
```

```
 PRIMARY KEY ( 'UID' )
);

*/

// nagłówek strony
echo '<html>
 <head>
  <title>Aktualności/title>
 </head>
 <body>
';

// łączenie z bazą danych
mysql_connect($db_host, $db_user, $db_pass);
mysql_select_db($db_database);

// funkcja wspierająca obsługę błędów bazy danych
function geterrdesc($sql) {
    $error = mysql_error() . "<BR>Error caused by statement: $sql";
    return $error;
}

// WYSWIETLANIE
if ($_GET['action']!="edit" && $_POST['action']!="save") {

if ($_GET['search'] != "") {
  $warunek = "WHERE ";
  $slowa = explode(" ", $_GET['search']);
  $ile_slow = count($slowa) - 1;
  $warunek .= "( ";
  for ($i=0; $i<$ile_slow; $i++)
   $warunek .= "instr(concat(body, ' ', title), '$slowa[$i]') <> 0 AND ";
 }

echo "<table cellpadding=\"4\" border=\"0\" cellspacing=\"1\" width=\"100%\">";
 $kolory[0]="#FFFFFF"; $kolory[1]="#CCCCCC"; $kolor=0;

if ($_GET[start]=="") $start=0; else $start=$_GET[start];

$sql = "SELECT NID,title,UID,time,body FROM news $warunek ORDER BY time DESC LIMIT $start, ⤸
 10";
 $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));
 while ($news = mysql_fetch_array($query)) {
  echo " <tr bgcolor=\"#333333\">
  <td width=\"15%\"> </td>
  <td width=\"85%\"><b><font color=\"#FFFFFF\" size=\"2\">" . $news[title] .
"</font></b></td>
 </tr>";

  $sql2 = "SELECT UID,username,status,obrazek,email,jabber FROM users WHERE UID=$news[UID]";
  $query2 = mysql_db_query($db_database, $sql2) or die(geterrdesc($sql2));
  $user = mysql_fetch_array($query2);

  echo " <tr bgcolor=\"$kolory[$kolor]\">
  <td width=\"15%\" valign=\"top\">
   <font size=\"2\"><b>" . $user[username] . "</b></font><br><center><img src=\"news_img/" .
⤸
   $user[obrazek] . "\" /><br><br><font size=\"1\">" . $user[status] .
"</font></center><br><⤸
```

```
   font size=\"1\">wyślij e-mail: <a href=\"mailto:" . $user[email] . "\"><img ⤸
   src=\"news_img/email.gif\" /></a></font><br> 
  </td>
  <td width=\"85%\" valign=\"top\"><font color=\"#000000\" size=\"2\">
   <table border=\"0\" width=\"100%\"><tr>
    <td><font size=\"1\">" . date('Y-m-d H:i:s O', $news[time]). "</font></td>
    <td align=\"right\"><a href=\"index.php?action=edit&NID=" . $news[NID] . "\"><img ⤸
    src=\"news_img/edit.gif\" /></a></td></tr></table><hr>
   " . $news[body] . "</font>
  </td>
 </tr>";
  if ($kolor==0) $kolor=1; else $kolor=0;
 }

echo "</table><p align=\"center\">";

if (($start-10) >= 0) echo "<a href=\"index.php?start=" . ($start-10) . "\">nowsze</a> ";
 $sql = "SELECT COUNT(*) FROM news";
 $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));
 $news = mysql_fetch_row($query);
 if ($news[0] > $start+10) echo "<a href=\"index.php?start=" . ($start+10) . "\">starsze</a>";

echo "</p>";
 echo "<form name=\"search_form\" method=\"GET\" action=\"index.php\"><p align=\"center\"><⤸
 input type=\"text\" name=\"search\" size=\"35\"><input type=\"submit\" value=\"Szukaj\"></p><⤸
 /form>";
 echo "<p align=\"center\"><a href=\"index.php?action=edit\">dodaj news'a</a></p>";

} elseif ($_GET['action']=="edit") {

if ($_GET[NID]=="") $NID=0; else $NID=$_GET[NID];

if ($NID !=0 ) {
  $sql = "SELECT NID,title,UID,body FROM news WHERE NID='$NID'";
  $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));
  $news = mysql_fetch_array($query);

  $sql = "SELECT UID,username FROM users WHERE UID='$news[UID]'";
  $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));
  $user = mysql_fetch_array($query);
 }

echo "<p><b>UWAGA:</b> Możliwość dodawania news'ów posiadają wyłącznie uprawnieniu ⤸
 użytkownicy posiadający konto w systemie.</p>";
 echo "<form name=\"news_form\" method=\"POST\" action=\"index.php\"><table>
 <tr><td>Temat:</td><td><input type=\"text\" name=\"title\" size=\"70\" value=\"$news[title]⤸
 \"></td></tr>
 <tr><td>Treść:</td><td><textarea name=\"body\" cols=\"70\" rows=\"14\">$news[body]<⤸
 /textarea></td></tr>";
 if ($NID==0) {
  echo " <tr><td>Autor:</td><td><input type=\"text\" name=\"autor\" size=\"35\"></td></tr>";
 } else {
  echo          " <tr><td>Usuń:</td><td><input type=\"checkbox\" name=\"delete\" value=\"yes\"
/></td>⤸
  </tr>";
  echo " <tr><td>Autor:</td><td>$user[username]</td></tr>";
 }
 echo "<tr><td>Hasło:</td><td><input type=\"password\" name=\"haslo\" size=\"35\"></td></tr>
 <tr><td colspan=\"2\" align=\"center\"><input type=\"hidden\" name=\"action\" value=\"save\">⤸
 <input type=\"hidden\" name=\"NID\" value=\"$NID\"><input type=\"submit\" value=\"Wyślij\"><⤸
```

```
 /td></tr>
</table></form>";

} else {

if ($_POST[NID]==0) {
  // wstawiamy nowy
  $sql = "SELECT UID,username,password FROM users WHERE username='$_POST[autor]'";
  $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));
  $user = mysql_fetch_array($query);

  if ($user[password] != $_POST[haslo] || $_POST[haslo]=="") {
   echo "<h2>Błędne hasło</h2></body></html>";
   exit;
  }

  $sql = "INSERT INTO news VALUES (NULL, '$_POST[title]', '$user[UID]', '" . time() . "', ⤸
  '$_POST[body]')";
  $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));

  echo "<h2>News został dodany</h2>";

} else {
  // aktualizujemy
  $sql = "SELECT NID,UID FROM news WHERE NID='$_POST[NID]'";
  $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));
  $news = mysql_fetch_array($query);

  $sql = "SELECT UID,password FROM users WHERE UID='$news[UID]'";
  $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));
  $user = mysql_fetch_array($query);

  if (($user[password] != $_POST[haslo] && $admin_pass != $_POST[haslo]) || $_POST[haslo]=="")⤸
  {
   echo "<h2>Błędne hasło</h2></body></html>";
   exit;
  }

  if ($_POST[delete]=="yes") {
   $sql = "DELETE FROM news WHERE NID='$_POST[NID]'";
   $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));

   echo "<h2>News został usunięty</h2>";
  } else {
   $sql = "UPDATE news SET title='$_POST[title]', body='$_POST[body]' WHERE NID='$_POST[NID]⤸
   '";
   $query = mysql_db_query($db_database, $sql) or die(geterrdesc($sql));

   echo "<h2>News został zmieniony</h2>";
  }
 }

}

echo "</body></html>";
?>
```

NutOS Jabber Notify Pack

Jest to zestaw klienta i serwera (opartego o mechanizm wyświetlania wiadomości wykorzystywany w bramce web), który pobiera z mysql dane o wiadomnościach jabbera i przsyła je do klienta - opartego na Mmnet02 i NutOSie. Okresowo wysyłany jest komunikat UDP o ilości wiadomości, natomiast po nawiązaniu połączenia przez klienta wysyłane są kolejne wiadomości.

POBIERZ[2179]

Więcej na temat Mmnet02 i NutOSie w układy programowalne.

O C API dla MySQL więcej można znaleźć w API MySQL C na LinuxFocus[2180]

JABBER_INFO_SERVER.C

```
/*
 * program stworzony w oparciu o program mojego autorstwa "jabber_view.c"
 * (pobierający i wyświetlający komunikaty IM sieci Jabber z bazy danych MySQL)
 * jest serwerem informacji dla (energooszczędnego) klijenta opartego na
 * mikrokontrolerze z interfejsem Ethernetowskim
 *
 * program wysyła okresowo komunikaty UDP z informacją o aktualnym czasie (UTC)
 * oraz liczba wiadomości przechowywanych dla danego użytkownika,
 * umożliwia on ponadto połączenie się po TCP celem pobrania treści przechowywanych ⤸
 wiadomości
 *
 *
 * program wywolany bez żadnych parametrów spróbuje pozyskać nazwę użytkownika
 * uruchamiającego i wykorzystać ją do pobrania danych o wiadomościach jabbera
 *
 * kompilacja: gcc jabber_info_server.c -o jabber_info_server -lmysqlclient -lpam -lpam_misc -⤸
 lpthread
 *
 */

// UWAGA: wkompilowane na stale do pliku binarnego haslo jest bardzo latwo z niego wyciagnac ⤸
!!!
// dlatego znacznie lepiej aby bylo ono w osobnym pliku do odczytu tylko dla uzytkownika
// na prawch ktorego chodzi ten program (z ustawionym suid - oczywiscie nie root)
// i aby bylo czytane z tego pliku konfiguracyjnego ...
//#define _haslo_do_bazy_danych_ ""
//#define _plik_z_haslem_do_bazy_danych_ "/etc/jabberd2/db_pass"

#define _SERWER_NAME_ "n17.waw.pl"
#define _PORT_ 3333
#define _PORT_UDP_ 4444
#define _IP_ "192.168.0.100"
#define BUF_SIZE 255
#define QUERY_SIZE 10

#include <security/pam_appl.h>
#include <security/pam_misc.h>
#include <mysql/mysql.h>
#include <stdio.h>

#include <pwd.h>
#include <unistd.h>
#include <sys/types.h>
#include <errno.h>

#include <pthread.h>
#include <sys/socket.h>
```

```
#include <netinet/ip.h>

/** zmienne globalne **/
char not_send_udp;
char buf[BUF_SIZE];
char query[100];
char query_udp[100];
#ifndef _haslo_do_bazy_danych_
 char _haslo_do_bazy_danych_ [255];
#endif


/** przetwarzanie wiadomosci zapisanej w XML **/
void parse_jabber_xml(char *napis0, char **parse_res){
 char *napis1, *napis2, *napis3;

// find time info
 napis2 = strstr(napis0, "stamp='");
 if (napis2 != NULL) {
  napis2 += 7;
  parse_res[3] = napis2;

  if ( strstr(napis1, ">") < napis2 )
   parse_res[0] = NULL;

  // find end from time stamp
  napis3 = strstr(napis2, "'");
 }

// find start from JID
 napis1 = strstr(napis0, "<message ");
 if (napis1 == NULL)
  return;
 napis2 = strstr(napis1, "from='");
 if (napis2 != NULL) {
  napis2 += 6;
  parse_res[0] = napis2;

  if ( strstr(napis1, ">") < napis2 )
   parse_res[0] = NULL;

  // find end from JID
  napis1 = strstr(napis2, "'");
  napis2 = strstr(napis2, "/");
  if (napis2<napis1) {
   *napis2 = '\0';
   napis0 = napis2 +1;
  } else {
   *napis1 = '\0';
   napis0 = napis1 +1;
  }
 }

// find start subject
 napis1 = strstr(napis0, "<subject>");
 if (napis1 != NULL) {
  napis1 += 6;
  parse_res[1] = napis1;

  // find end subject
```

```
  napis1 = strstr(napis1, "</subject>");
  *napis2 = '\0';
  napis0 = napis2 + 1;
 }

// find start message
 napis1 = strstr(napis0, "<body>");
 if (napis1 != NULL) {
  napis1 += 6;
  parse_res[2] = napis1;

  // find end message
  napis1 = strstr(napis1, "</body>");
  *napis1 = '\0';
 }

// mark end of time
 *napis3 = '\0';
}

/** wątek odpowiedzialny z wysyłanie informacji UDP **/
int send_udp_info() {
 char udp_buf[36];
 time_t timestamp;
 struct tm czas;

int sh_udp = socket(PF_INET, SOCK_DGRAM, 0);
 if (sh_udp<0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }
 struct sockaddr_in to;
 to.sin_family=PF_INET;
 to.sin_port=htons(_PORT_UDP_);
 inet_aton(_IP_, &to.sin_addr);

/** pobieranie dancyh **/
while (1) {
  MYSQL mysql_connection_udp;
  MYSQL_RES *result_udp;
  MYSQL_ROW row_udp;
  mysql_init(&mysql_connection_udp);
  mysql_real_connect(&mysql_connection_udp, "localhost", "jabberd2",
   _haslo_do_bazy_danych_, "jabberd2", 3306, NULL, 0);

  mysql_real_query(&mysql_connection_udp, query_udp, strlen(query_udp));
  result_udp = mysql_use_result(&mysql_connection_udp);
  row_udp=mysql_fetch_row(result_udp);

  /** czas i wysylka **/
  time(&timestamp);
  gmtime_r(&timestamp, &czas);

  sprintf(udp_buf, "%.4d-%.2d-%.2d %.2d:%.2d\n%s wiadomosc%s",
   czas.tm_year+1900, czas.tm_mon+1, czas.tm_mday, czas.tm_hour, czas.tm_min,
   row_udp[0], atoi(row_udp[0])==1?"":"i");
  //puts(udp_buf);
  sendto(sh_udp, udp_buf, strlen(udp_buf), 0, (struct sockaddr *) &to, sizeof(struct
  sockaddr_in));
```

```
  mysql_close(&mysql_connection_udp);

  sleep(60);
  while(not_send_udp)
   sleep(60);
 }
}

int main(int argc, char *argv[]) { // pierwszy argument - uzytkownik , STDIN - haslo
 char * username;
 struct passwd *userstr;

if (argc == 2) {
  /** autoryzacja użytkownika **/

  username = argv[1];

  struct pam_conv conv = {misc_conv, NULL};
  pam_handle_t *pam = NULL;

  int pam_connection = pam_start("jabberd", username, &conv, &pam);

  if (pam_authenticate(pam, 0) != PAM_SUCCESS) {
   fprintf(stderr, "Błąd autoryzacji użytkownika\n");
   return -1;
  } else fprintf(stderr, "OK\n");

  pam_end(pam, pam_connection);
 } else {
  /** pobieranie nazwy aktualnego uzytkownika **/

  userstr = getpwuid( getuid() );
  if (userstr)
   username = userstr->pw_name;
 }

sprintf(query,
  "SELECT `xml` FROM `queue` WHERE `collection-owner`='%s@%s' ORDER BY `object-sequence`",
  username, _SERWER_NAME_);
 sprintf(query_udp,
  "SELECT COUNT(*) FROM `queue` WHERE `collection-owner`='%s@%s'",
  username, _SERWER_NAME_);

/** polaczenie i obsluga MySQL **/

#ifndef _haslo_do_bazy_danych_
 FILE *plik;
 plik=fopen(_plik_z_haslem_do_bazy_danych_, "r");
 fgets( _haslo_do_bazy_danych_, 254, plik );
 int ostatni = strlen(_haslo_do_bazy_danych_) - 1;
 _haslo_do_bazy_danych_[ostatni]='\0';
#endif


/** wątek informacji UDP **/
pthread_t watek1;
 not_send_udp = 0;
 pthread_create( &watek1, 0, send_udp_info, NULL );
```

```
/** nasłuchiwanie TCP **/
int res;

// gniazdo ...
 int sh = socket(PF_INET, SOCK_STREAM, 0);
 if (sh<0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }

// utworzenie struktury opisującej adres
 struct sockaddr_in serwer;
 serwer.sin_family=AF_INET;
 serwer.sin_port=htons(_PORT_);
 serwer.sin_addr.s_addr=INADDR_ANY;

// przypisanie adresu ...
 if (bind(sh, (struct sockaddr *) &serwer, sizeof(struct sockaddr_in)) < 0) {
  fprintf(stderr, "%s\n", strerror(errno));
  exit(EXIT_FAILURE);
 }

while(1) {
  // otwarcie portu do nasluchiwania
  if (listen(sh, QUERY_SIZE) < 0) {
   fprintf(stderr, "%s\n", strerror(errno));
   exit(EXIT_FAILURE);
  }

  // odebranie połączenia
  struct sockaddr_in from;
  socklen_t fromlen=sizeof(struct sockaddr_in);
  int sh2 = accept(sh, (struct sockaddr *) &from, &fromlen);
//    not_send_udp = 1;

  if (from.sin_addr.s_addr != inet_addr(_IP_)) {
   close(sh2);
   continue;
  }

  FILE * net;
  net=fdopen(sh2, "r+");

  fgets(buf, BUF_SIZE, net);
  if (buf[0] == 'n') {
   MYSQL mysql_connection;
   MYSQL_RES *result;
   MYSQL_ROW row;
   mysql_init(&mysql_connection);
   mysql_real_connect(&mysql_connection, "localhost", "jabberd2",
    _haslo_do_bazy_danych_, "jabberd2", 3306, NULL, 0);

   mysql_real_query(&mysql_connection, query, strlen(query));
   result = mysql_use_result(&mysql_connection);
   do {
    /** pobieranie, przetwarzanie i wysylanie dancyh **/
    char *parse_res[4], *czas;
    if (row=mysql_fetch_row(result)) {
     parse_res[0] = NULL; parse_res[1] = NULL; parse_res[2] = NULL; parse_res[3] = NULL;
     parse_jabber_xml(row[0], parse_res);
```

```
      czas = parse_res[3];
      czas[14] = '\0';
      if (parse_res[1] != NULL) {
       fprintf(net, "%s %s [%s] %s", czas+9, parse_res[0], parse_res[1], parse_res[2]);
      } else {
       fprintf(net, "%s %s %s", czas+9, parse_res[0], parse_res[2]);
      }
      fflush(net);
     } else {
      fputc(0x04, net); // EOT
      fflush(net);
      break;
     }
     fgets(buf, BUF_SIZE, net);
    } while (buf[0] == 'n');

   mysql_close(&mysql_connection);
  }

  fclose(net);
  not_send_udp = 0;
 }

// zamkniecie gniazda
 close(sh);
}
```

JABBER_INFO.C

```
/*
 * Program przeznaczony dla systemu Nut/OS ( http://www.ethernut.de/ ), służy do ⤸
 komunikacji
 * mikrokontrolera z serwerem informacji o wiadomościach IM sieci Jabber;
 * liczbę oczekujących wiadomości otrzymuje jako regularnie wysyłany pakiet UDP
 * oraz umożliwia przejrzenie pełnych treści wiadomości z wykorzystaniem protokołu TCP
 *
 * FUNKCJE PRZYCISKOW:
 * SW0 - wyswietl (nastepna) wiadomosc; gdy:
 *          wyswietlany komunikat, oczekiwanie na komunikat, informacja o koncu wiadomosci
 * SW0 - wyjscie do trybu komunikatów; gdy:
 *          wyswietlane info ze nie ma wiecej wiadomosci
 * SW1 - wyjscie do trybu komunikatów; gdy:
 *          informacja o koncu wiadomosci, wyswietlane info ze nie ma wiecej wiadomosci
 * SW1 - wyjscie do "koniec wiadomosci"; gdy:
 *          wyswietlana wiadomosc
 * SW2 / SW3 - przewijanie wiadomosci w przdd / tyl; gdy
 *          wyswietlana wiadomosc
 *
 */

/*
 */


#include <stdio.h>
#include <io.h>
#include <sys/timer.h>
```

```
// UART and Ethernet and LCD
#include <dev/board.h>
#include <dev/nicrtl.h>
#include <arpa/inet.h>
#include <sys/confnet.h>
#include <dev/hd44780_bus.h>
#include <dev/term.h>

#include <sys/socket.h>

#define _UDP_PORT_ 4444

// bezposredni dostep do pamieci po adresie, w oparciu o funkcje nic_inlb, nic_outlb z
ethernut
#define mem_read_byte(reg) (*(volatile u_char *) (reg))
#define mem_write_byte(reg, val) (*(volatile u_char *) (reg) = (val))


// funkcja pobiera linię z UART
// jezeli podano out != NULL linia bedzie wypisana do tego pliku (np. na uart ...)
void UartGetLine(u_char *buff, int len, FILE *out) {
 int i=0;
 while (i<len) {
  while ((inb(USR) & 1<<RXC) == 0); // czekamy na znak
  buff[i] = inb(UDR);
  if (out!= NULL) fputc(buff[i], out);
  if (buff[i] == '\n')
   break;
  i++;
 }
 buff[i] = '\0';
}

// funkcja pobierajaca konfiguracje IP z UART
u_long UartGetConfig(u_char * msg, u_long val, FILE * out) {
 u_long tmp;
 char inbuff[32];
 while(1) {
  fprintf(out, "Podaj %s [%s]: ", msg, inet_ntoa(val));
  UartGetLine(inbuff, 32, out);
  if (inbuff[0]=='\0' || inbuff[1]=='\0') {
   fprintf(out, " ... bez zmian\n");
   return val;
  }
  tmp=inet_addr(inbuff);
  if (tmp != (u_long) -1) {
   fprintf(out, " ... zmieniono\n");
   return tmp;
  }
 }
}




void main(void) {
 /// konfiguracja trybu pracy kontrolera pamięci
 mem_write_byte(0xff01, 0x00);
 /// wylaczenie przetwornikow A/D
```

```
 sbi(ACSR, ACD);


/// uruchamianie RS-232
 NutRegisterDevice(&DEV_UART, 0, 0);
 outb(UCR, 1<<RXEN | 1<<TXEN); //| 1 << RXCIE; // włączenie nadajnika, odbionika UART wraz ⤸
 z generowaniem przerwań
 outb(UBRR, 51); // szybkosc transmisji 9600 bps
 FILE *uart = fopen(DEV_UART_NAME, "w");

/// wypisanie czegos przez RS-232
 fprintf(uart, "Hello World !!!\n");
 fprintf(uart, "connection detect ...\n");

/// wykrywanie polaczenia przez RS-232
 uint32_t i;
 char uart_is_use=0;
 for (i=0; i<2000000; i++) { // ok 4 sek.
  if ((inb(USR) & 1<<RXC) != 0) {
   uart_is_use=1;
   break;
  }
 }

if (!uart_is_use) {
  fprintf(uart, "UART is NOT use !\n");
 } else {
  fprintf(uart, "UART is use !\n");

  /// oproznienie bufora
  while ((inb(USR) & 1<<RXC) != 0) inb(UDR);

  fprintf(uart, "Napisz cos:\n");

  /// wczytanie czegos przez RS-232
  char inbuff[32];
  UartGetLine(inbuff, 32, uart);
  fprintf(uart, "Napisales: %s\n", inbuff);
 }


/// konfigurator sieci
 fprintf(uart, "Konfigurowanie sieci\n");
 NutNetLoadConfig("eth0");
 u_long ip_add=confnet.cdn_cip_addr;
 u_long ip_mask=confnet.cdn_cip_addr;
 if (ip_add==0) {
  fprintf(uart, "  uzycie konfiguracji domyslnej\n");
  ip_add=0x6418A8C0; // 192.168.0.100
  ip_mask=0x0000ffff; // 255.255.0.0
 }

if (uart_is_use) {
  ip_add = confnet.cdn_cip_addr = UartGetConfig("IP  ", ip_add, uart);
  ip_mask = confnet.cdn_ip_mask = UartGetConfig("Mask", ip_mask, uart);
  fprintf(uart, "zapisac [y/*]\n");
  char inbuff[3];
  UartGetLine(inbuff, 3, uart);
  if (inbuff[0]=='y') {
   fprintf(uart, " zapisano\n");
```

```
   NutNetSaveConfig();
  }
 }


/// siec
 fprintf(uart, "Wczytana konfiguracja sieci:\n");
 fprintf(uart, "IP:     %s (%lx)\n", inet_ntoa(ip_add), ip_add);
 fprintf(uart, "Mask:   %s (%lx)\n", inet_ntoa(ip_mask), ip_mask);
 //jest tez: confnet.cdn_gateway oraz confnet.cdn_mac[0] ... confnet.cdn_mac[5]

fprintf(uart, "Uruchamianie sieci\n");
 if (NutRegisterDevice(&DEV_ETHER, 0x8300, 5))
  fputs("Registering device failed", uart);

if(NutNetIfConfig("eth0", NULL, ip_add, ip_mask))
 fprintf(uart, "Siec skonfigurowana\n");


/// lcd
 fprintf(uart, "LCD Start\n");
 NutRegisterDevice(&devLcdBus, 0xff04, 0);
 FILE *lcd = fopen("lcdbus", "w");
 fprintf(lcd, ESC_CLR "oczekiwanie na\nkomunikat");

DDRD = 0;
 PORTD = 0xff;

UDPSOCKET* sock;
 sock = NutUdpCreateSocket(_UDP_PORT_);
 TCPSOCKET* sock_tcp;
 u_long addr;
 u_short port;
 u_char data[225];
 u_short size;
 u_char set_break;


while (1) {
  while ((PIND & 0x01) != 0) {
   size = NutUdpReceiveFrom(sock,&addr,&port,data,sizeof(data),100);
   if (size > 0) {
    data[size]=0;
    fprintf(uart, "%i : %s", size, data);

    fprintf(lcd, ESC_CLR);
    fprintf(lcd, "%s", data);
   }
  }
  while ((PIND & 0x01) == 0);

  // po nacisnieciu przysisku

  sock_tcp = NutTcpCreateSocket();
  NutTcpConnect(sock_tcp, inet_addr("192.168.0.1"), 3333);
  set_break = 0;

  while (1) { // do nacisniecia guzika
   NutTcpSend(sock_tcp, "n\n", 2);
   do {
```

```
      size = NutTcpReceive(sock_tcp, data, sizeof(data)-1);
      data[size]='\0';
      fprintf(uart, "%i : %s", size, data);
      if (data[0]==0x04) {
       fprintf(lcd, ESC_CLR "Nie ma wiecej\nwiadomosci");
       set_break = 1;
      } else {
       // wyswietlanie z podzialem na linie
       i = 0;
       while(1) {
        if (i<0)
         i = 0;

        if (i+16<size)
         fprintf(lcd, ESC_CLR "%.16s\n%.16s", data+i, data+i+16);
        else if (i<size)
         fprintf(lcd, ESC_CLR "%.16s", data+i);
        else
         break;

        while ((PIND & 0x04) != 0 && (PIND & 0x08) != 0 && (PIND & 0x02) != 0);
         // czekamy na PD1, PD2 lub PD3 zwarte do masy
        if ((PIND & 0x04) == 0)
         i=i+32;
        else if ((PIND & 0x08) == 0)
         i=i-32;
        else if ((PIND & 0x02) == 0)
         break;

        NutSleep(100);
        while ((PIND & 0x04) == 0 || (PIND & 0x08) == 0 || (PIND & 0x02) == 0);
         // wszystkie 3 w stanie wysokim
       }
       if (size < 224)
        fprintf(lcd, ESC_CLR "koniec\nwiadomosci");
      }
     } while (size >= 224);

     while ((PIND & 0x01) != 0 && (PIND & 0x02) != 0);
      // czekamy na PD0 lub PD1 zwarte do masy
     if ((PIND & 0x02) == 0 || set_break == 1)
      break;

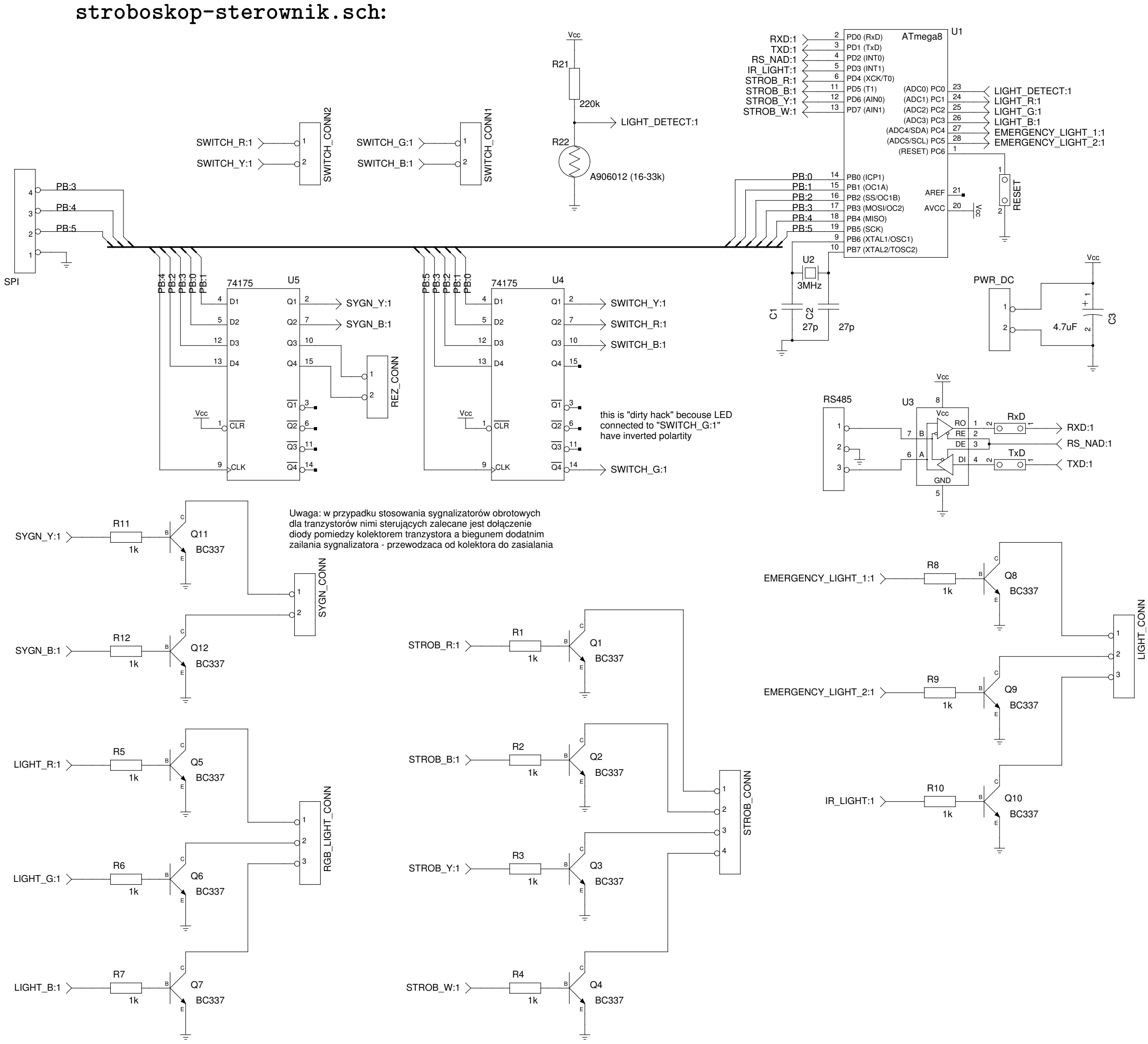
     NutSleep(100);
     while ((PIND & 0x01) == 0 || (PIND & 0x02) == 0);
      // oba w stanie wysokim
    }
    NutTcpCloseSocket(sock_tcp);
    fprintf(lcd, ESC_CLR "oczekiwanie na\nkomunikat");
  }
}
```

Stroboskop LED

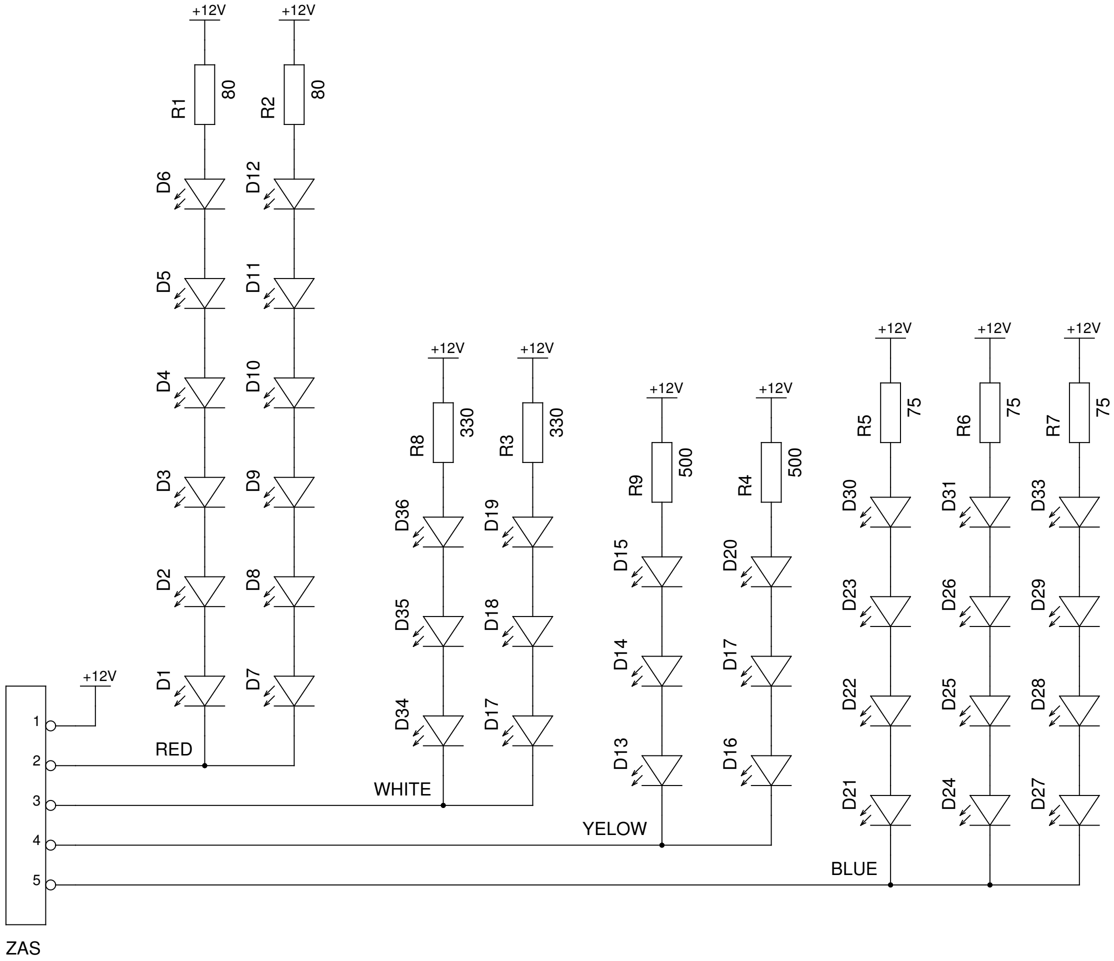
Jest to projekt 4-kolorowego stroboskopu opartego na LED oraz mikrokontrelerze Atmega8. Stworzony został on na potrzeby sygnalizacji w systemie alarmowym. Dodatkowo niewykorzystane porty mikrokontrelera zostały użyte do sterowania innymi urządzeniami (oświetlenie awaryjne, sygnalizatory alarmowe, itp.)

- program sterujący praca mikrokontrolera (Wymagane są następujące nagłówki - wykorzystywane także w innych projektach: my_rs485_ascii.c)
- schemat ideowy płytki sterownika[2181](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-sterownik.pdf`)
- projekt płytki drukowanej[2182](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-sterownik.pcb.pdf`)
- schemat ideowy matrycy FLUX LED zasilanej z 12V[2183](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-matryca_12V.pdf`)
- schemat ideowy matrycy FLUX LED zasilanej z 5V[2184](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-matryca_5V.pdf`)
- koncepcja realizacji matrycy z FLUX LED[2185](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/matryca_flux.pcb.pdf`)

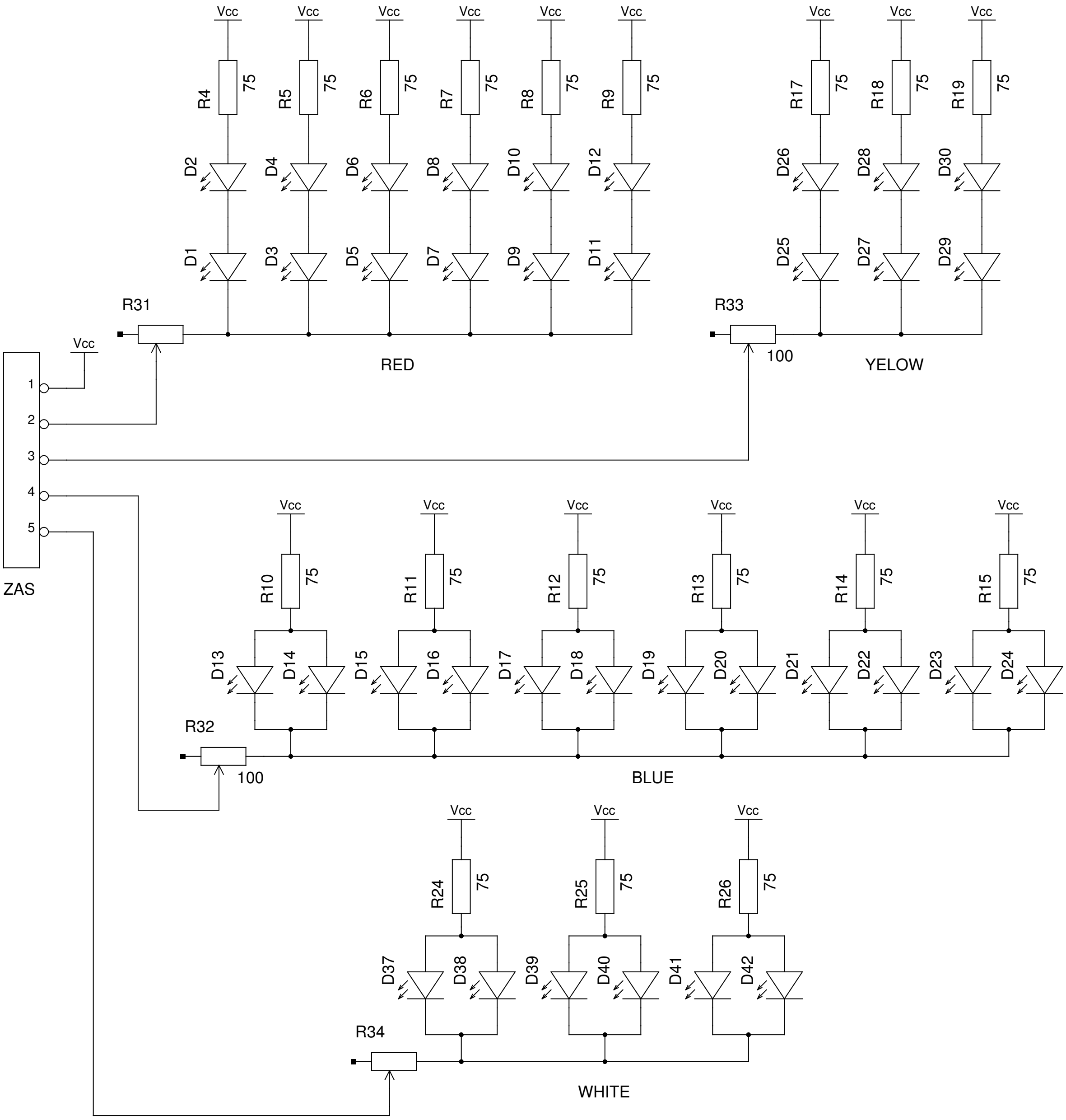
stroboskop-sterownik.sch:



stroboskop-matryca_12V.sch:



stroboskop-matryca_5V.sch:



Pobierz[2186]

STROBOSKOP-STEROWNIK.C

```
/*
 * program sterownika stroboskopu i kluczy tranzystorowych
 * z funkcją zdalnego sterowania poprzez port RS
 * dla mikrokontrolera Atmega8
 *
 *
 * kompilacja i wgranie przez ISP:
 *     avr-gcc -mmcu=atmega8 -Os -o sterownik.o stroboskop.c
 *     avr-objcopy -O ihex sterownik.o sterownik.hex
 *     uisp -dprog=dapa -dlpt=0x378 --erase --upload --verify --segment=flash if=sterownik.hex
 *     w razie problemow warto przeladowac modul odpowiedzialny za obsluge LPT
 *         rmmod lp parport_pc parport ; modprobe parport_pc
 *
 * konfiguracja układu na zewnętrzny kwarc 8MHz (lub troche mniejszy)
```

```
 *    uisp -dprog=dapa -dlpt=0x378 --wr_fuse_l=0xff
 */

/** MATRYCA LED 12x3:
 R R R R Y W W Y B B B B
 R R R R W Y Y W B B B B
 R R R R Y W W Y B B B B

 LUB:

 R R R R Y W Y B B B B
 R R R R W Y W B B B B
 R R R R Y W Y B B B B

 R = dioda czerwona (PD.4)
 B = dioda niebieska (PD.5)
 Y = dioda żółta (PD.6)
 W = dioda biała (PD.7)

 podane piny portu D służą do podłączania tranzystorów sterujących diodami
 **/

// biala, niebieska 0.5k jedna w szereg, dwie rownolegle
// zolta, czerwona 0.33k dwie w szereg

#include <inttypes.h>
#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>

#define STROBOSKOP

/** nagłówki funkcji opóźniających **/
#define F_CPU 3000000UL
#include <util/delay.h>

/** nagłówki mojego formatu transmisji przez RS485 **/
#define DRIVER_CTRL_PORT PORTD
#define DRIVER_CTRL_PIN PD2
#define MY_ADDRES 22

#include "my_rs485_ascii.c"


/** struktura z polami bitowymi odpowiadajacymi kolorm ... **/
volatile struct {
 uint8_t r :1;
 uint8_t b :1;
 uint8_t y :1;
 uint8_t w :1;
} kolory;

/// maski bitowe przechowujące informację o stanie linii
#define STATUS_MAX 5
volatile uint8_t status [STATUS_MAX];
//    status[0] - PORTD
//    status[1] - PORTC
//    status[2] - rejestr U4 (aktywowanego PB4)
//    status[3] - rejestr U5 (aktywowanego PB5)
//    status[STATUS_MAX-1]
```

```
//                    - 4 młodsze bity - zabroń migania dla kolejnych kolorów
//                    - 4 starsze bity - zabroń stroboskopu dla kolejnych kolorów
//                       (ze względów technicznych powoduje to równierz wyłączenie miganie))
//                    bit 0 -> R, bit 3 -> W, bit 4 ->R , bit 7 -> W
//                    ustawienie na 0xff lub 0xf0 spowoduje stałe świecenie diód stroboskopu

void make_strobe(uint8_t mask){
 _delay_us(30.0);
 PORTB |= mask;
 _delay_us(60.0);
 PORTB &= 0x0f;
}

/** obsluga portu szeregowego **/
// subadres:
//     0 -> IR_LIGHT:1
//     1 -> STROB_R:1
//     2 -> STROB_B:1
//     3 -> STROB_Y:1
//     4 -> STROB_W:1
//
//     5 -> LIGHT_R:1
//     6 -> LIGHT_G:1
//     7 -> LIGHT_B:1
//     8 -> EMERGENCY_LIGHT_1:1
//     9 -> EMERGENCY_LIGHT_2:1
//
//    14 -> SWITCH_Y:1
//    15 -> SWITCH_R:1
//    16 -> SWITCH_B:1
//    17 -> SWITCH_G:1
//
//    10 -> SYGN_B:1
//    11 -> SYGN_Y:1
//    12 -> (REZERWA)
//    13 -> (REZERWA)

void execute_rs_cmd(uint8_t cmd, uint8_t addr, uint8_t param, uint8_t uart_decode_mode) {
 uint8_t addr_ind;
 if (addr > 17) {
  addr_ind = 6;
  addr = addr - 18;
 } else if (addr > 13) {
  addr_ind = 3;
  addr = addr - 14;
 } else if (addr > 9) {
  addr_ind = 2;
  addr = addr - 10;
 } else if (addr > 4) {
  addr_ind = 1;
  addr = addr - 5;
 } else {
  addr_ind = 0;
 }
 switch (cmd) {
  case 0x01: /// read
   if ( (status[addr_ind] & (1 << addr)) != 0 )
    rs_switch_and_send_char('1'); // ON
   else
    rs_switch_and_send_char('0'); // OFF
```

```
      break;
    case 0x02: /// on
      status[addr_ind] = status[addr_ind] | (1 << addr);
      break;
    case 0x03: /// off
      status[addr_ind] = status[addr_ind] & ~(1 << addr);
      break;
    case 0x04: /// switch
      status[addr_ind] = status[addr_ind] ^ (1 << addr);
      break;
    case 0x05: /// get status
      if (addr < STATUS_MAX) {
        rs_switch_and_send_8bit_num( status[addr] );
      }
      break;
    case 0x06: /// set status
      if (addr < STATUS_MAX) {
        status[addr] = param;
        addr_ind = addr;
      }
      break;
    case 0x07: /// odczytaj napięcie związane z fotorezystorem
          //   odczytaną wartość należy pomnożyć przez 0.1V
          //   maksymalny odczyt to 2.55V
      // pomiar ADC
      ADCSRA |= _BV(ADSC);
      // oczekiwanie na zakończenie pomiaru
      while(bit_is_set(ADCSRA,ADSC));
      // wysłanie wyniku
      rs_switch_and_send_8bit_num( ADCH );
      break;
  }
  switch (addr_ind) {
    case 3:
      PORTB = status[3] & 0x0f;
      make_strobe(0x20);
      break;
    case 2:
      PORTB = status[2] & 0x0f;
      make_strobe(0x10);
      break;
    case 1:
      PORTC = status[1] << 1;
      break;
    case 0:
      PORTD &= 0x07;
      PORTD |= (status[0] << 3 )& 0xf8;
      break;
  }
}


#ifdef STROBOSKOP
#define TCNT0_INIT_20ms 256 - (19 * F_CPU / 1000000UL)
uint8_t licznik_timer_20ms = 0;
SIGNAL (SIG_OVERFLOW0) {
  // reinicjalizujemy timer
  TCNT0 = TCNT0_INIT_20ms;
```

```
licznik_timer_20ms++;

if (licznik_timer_20ms%8 == 0) {
  // włączamy kolory żółty i biały pod warunkiem że akurat świecą
  PORTD |= kolory.y << 6 | kolory.w << 7;
 } else if (licznik_timer_20ms%4 == 0) {
  // wyłączamy kolory żółty i biały
  PORTD &= (0x3f | status[STATUS_MAX-1]);
 }

if (licznik_timer_20ms%10 == 0) {
  // włączamy kolory czerwony i niebieski pod warunkiem że jest zezwolenie w kolory
  PORTD |= kolory.r << 4 | kolory.b << 5;
 } else if (licznik_timer_20ms%5 == 0) {
  // wyłączamy kolory czerwony i niebieski
  PORTD &= 0xcf | status[STATUS_MAX-1];
 }

if (licznik_timer_20ms == 80)
  licznik_timer_20ms = 0;
}
#endif


main() {
 DDRD = 0xfe;
 status[0]=0x00; // 0x1e -> stroboskop wlaczony
 PORTD = 0;

DDRC = 0xfe;
 status[1]=0x00;
 PORTC = 0;

DDRB = 0xff;
 status[2]=0x00;
 PORTB = 0;
 make_strobe(0x10);

status[3]=0x00;
 PORTB = 0;
 make_strobe(0x20);

kolory.r=0;
 kolory.b=1;
 kolory.y=1;
 kolory.w=0;

/** inicjalizacja ADC **/
// wewnętrzne źródło odnienienia, interesuje nas 8 starszych bitów konwersja z ADC0 (
 PC0)
 ADMUX |= _BV(REFS0) | _BV(REFS1) | _BV(ADLAR);
 // aktywacja orazustawienie częstotliwości
 ADCSRA |= _BV(ADEN) | _BV(ADPS0) | _BV(ADPS1);

/** port szeregowy - inicjalizacja **/
rs_init();


#ifdef STROBOSKOP
 // aktywujemy timer dizałający co 20ms
```

```
 TIMSK = _BV(TOIE0); // timer generuje przerwania przy przepełnieniu
 TCNT0 = TCNT0_INIT_20ms; // wartość początkowa timera
 TCCR0 = _BV(CS00)|_BV(CS02); // dzielinik zegara systemowego 1024
 sei(); // właczamy globalną obsługe przerwań
#endif

uint8_t licznik_timer_100ms = 0;
 while(1) {
  if (licznik_timer_100ms%19 == 0) {
   // YELOW ON
   kolory.y=(status[0] >> 3) & 0x01;
  } else if ((licznik_timer_100ms-5)%19 == 0) {
   // YELOW OFF
   kolory.y=(status[STATUS_MAX-1] >> 2) & 0x01;
  } else if ((licznik_timer_100ms-8)%19 == 0) {
   // WHITE ON
   kolory.w=(status[0] >> 4) & 0x01;
  } else if ((licznik_timer_100ms-13)%19 == 0) {
   // WHITE OFF
   kolory.w=(status[STATUS_MAX-1] >> 3) & 0x01;
  }

  if (licznik_timer_100ms%12 == 0) {
   // RED OFF
   kolory.r=(status[STATUS_MAX-1]) & 0x01;
   // BLUE ON
   kolory.b=(status[0] >> 2) & 0x01;
  } else if (licznik_timer_100ms%6 == 0) {
   // RED ON
   kolory.r=(status[0] >> 1) & 0x01;
   // BLUE OFF
   kolory.b=(status[STATUS_MAX-1] >> 1) & 0x01;
  }

  if (++licznik_timer_100ms == 180) {
   licznik_timer_100ms = 0;
  }

#ifndef STROBOSKOP
#warning nie zdefiniowano STROBOSKOP - program tylko dla celow testowych - moze byc zaburzone ⤸
sterowanie nadajnikiem RS485
  // wyświetlenie wyjścia bez nałożenia efektu stoboskopowego
  PORTD = kolory.r << 4 | kolory.b << 5 | kolory.y << 6 | kolory.w << 7;
  // jeżeli zachowamy porządek bitów zgodny z kolory to kompilator
  // to ładnie sobie zoptymalizuje
#endif

  // czekamy około 100ms
  uint8_t i=0;
  for (i=0; i<5; i++) _delay_ms(20.0);
 }
}
```

Sterownik ogrzewania akwariowego

Na stronie tej zamieszczam projekt oraz oprogramowanie mikroprocesorowego sterownika ogrzewania akwariowego, opartego na mikrokontrolerze AT90S2313, wykonanego w 2005 roku. Układ posiada dwa niezależne tory sterownicze ("akwarium" i "terrarium") umożliwiające sterowanie dwoma zestawami grzałek z wykorzystaniem dwóch termometrów cyfrowych DS18b20. Informacje o temperaturze i jej odniesieniu do zadanych parametrów podawane są na wyświetlaczu alfanumerycznym LCD (opartym o sterownik HD44780). System umożliwia komunikację (odczyt i zapis konfiguracji oraz odczyt aktualnych parametrów i sterowanie urządzeniem) poprzez port RS232.



Elektronika

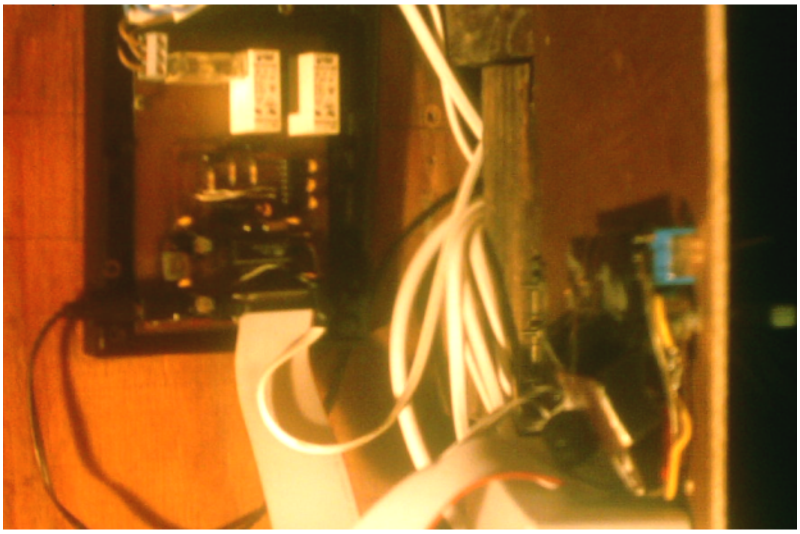
Część elektroniczna składa się z płytki drukowanej, na której umieszczone zostały moduł zasilania (stabilizator +5V), moduł mikrokontrolera, moduł komunikacji z RS232 oraz moduł wykonawczy (przekaźniki) i podłączanej do niej 26 pinową taśmą płytki czołowej na której znajduje się wyświetlacz, złącze RS232, reset i sygnalizator alarmu.

schemat płytki obwodów elektronicznych[2187](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/sterownik_ogrzewania/mikrokontroler.pdf`)
projekt płytki drukowanej PCB[2188](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/sterownik_ogrzewania/mikrokontroler.pcb.pdf`)
podłączenia płytki czołowej[2189](PDF: `http://www.opcode.eu.org//usage_and_config/other_projects/sterownik_ogrzewania/plytka_czolowa.pdf`)

Oprogramowanie

Oprogramowanie składa się z programu głównego dla mikrokontrolera (mikro.c) dwóch bibliotek funkcji (lcd.c i onewire.c) do obsługi wyświetlacza i termometru oraz programu do odczytu numeru ROM termometru (onewire_get_rom.c). W poniższym archiwum zamieszczone dodatkowo są także wersje PC'owe programów do obsługi termometru i wyświetlacza.

Fotki





Pobierz[2190]

Kierunki rozwoju

W przyszłości można by zastąpić RS232 poprzez RS485, co umożliwiłoby pracę sterownika w sieci. Rozważyć można także zmianę protokołu komunikacji RS z binarnego na tekstowy, a także zastąpienie przekaźników zestawem triak + optotriak. Możliwe byłoby dodanie do sterownika (raczej po przeportowaniu na Atmega8) funkcji dwu kanałowego sterownika oświetlenia.

Zobacz w Sieci: Wielozadaniowy sterownik akwarystyczny "MTerm"[2191], Skalar - Mikroprocesorowy Sterownik Akwariowy[2192], Sterownik do akwarium[2193].

mikro.c

```
/**

 * program realizujący sterownik ogrzewania akwariowego na bazie mikrokontrolera AT90S2313

 * kompilacja i wgranie przez ISP:
 *     avr-gcc -mmcu=at90s2313 -Os -o mikro.o mikro.c
 *     avr-objcopy -O ihex mikro.o mikro.hex
 *     uisp -dprog=dapa -dlpt=0x378 --erase --segment=flash
 *     uisp -dprog=dapa -dlpt=0x378 --upload --segment=flash if=mikro.hex
 *     uisp -dprog=dapa -dlpt=0x378 --verify --segment=flash if=mikro.hex

**/


/** nagłówki ogólne **/

#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/signal.h>
#include <avr/eeprom.h>

#define bool unsigned char
#define sti() sei()

/*
long long czas() { asm( "rdtsc" ); }
void opoznienie( long long l ) {
 l = l * 3012 + czas(); // prędkość procesora w MHz
 while( l > czas() );
```

```
}
#define opoznienie(czas) opozniacz(czas * 0.67)
void opozniacz (long l) {
 for (; l; --l ) asm volatile( "" );
}*/

#define F_CPU 7372800UL
#include <util/delay.h>

/** nagłówki LCD **/

#define lcd_send(byte) PORTB = (PORTB | 0x3f) & (byte & 0x3f)
#include "lcd.c"


/** nagłówki OneWire **/

#define onewire_send_zero()  DDRB |= 0x40; PORTB &= 0xbf
#define onewire_send_jeden() PORTB |= 0x40; DDRB &= 0xbf
#define onewire_read()       (PINB & 0x40) >> 6
#include "onewire.c"


/** zmienne globalne **/

unsigned char ustawienia[28];/*={0, 0, 22, 25, 18, 33,\
       0x28, 0x46, 0xba, 0x89, 0x0, 0x0, 0x0, 0x61,\
                               0, 0, 22, 25, 18, 33,\
       0x28, 0x31, 0x8f, 0x89, 0x0, 0x0, 0x0, 0x2a};*/
unsigned char temperatura_ascii[24];
unsigned char temp, temp_old, dn, t0, t1, j;
char alarm[2];


/** port szeregowy **/

char uart_buf, uart_set;
SIGNAL( SIG_UART_RECV ) {
 cli();

uart_buf = UDR;

if (uart_set != -1) {
  ustawienia[ uart_set ] = uart_buf;
  uart_set = -1;
  UDR = 'e'; // end set mode
  goto END;
 }

uart_set = (uart_buf & 0x0f) + ((uart_buf >> 4) & 0x01) * 14;

switch (uart_buf & 0xc0) {
  case 0x40:
   UDR = ustawienia[ uart_set ];
   uart_set = -1;
   break;
  case 0x80:
   UDR = 's'; // start set mode
   break;
  case 0xc0:
```

```
    for (j=0; j<28; j++) {
     eeprom_write_byte(j, ustawienia[j]);
     eeprom_busy_wait();
    }
    uart_set = -1;
    UDR = 'o'; // ok
    break;
  }
END:
  sti();
}
/*
starsze 4 bity - okreslenie funkcji (get/set/save), urzadzenia
 funkca | urzadzenie
 funkcja:
   0x4 - get
   0x8 - set
   0xc - save
 urzadzenie
   numer urzadzenia od 0x0 do 0x1 (maksymalnie 0x3)
młodze 4 bity - numer wartosci:
 0x0 - temperatura
 0x1 - granica dolna
 0x2 - granica górna
 0x3 - granica alarmu dolna
 0x4 - granica alarmu górna
 0x5 - grzanie (dwa najstarsze bity: 00 = auto, 10 = on, 01 = off)
   // poprzecinkowa czesc temperatury (4 najmlodsze bity)
 0x6 - ROM (0)
 ...
 0xd - ROM (7)
*/
/*
 UCR = 1 << TXEN | 1 << RXEN | 1 << RXCIE;
   // włączenie nadajnika, odbionika UART wraz z generowaniem przerwań
 UBRR = 47; // szybkosc transmisji 9600 bps
 uart_set = -1;
 sti(); // włączenie przerwań
*/


/** sterowanie grzalkami i alarmem **/

void grzanie(unsigned char device, bool on) {
  if (ustawienia[ 5 + 14*dn ] & 0x80) {
   on=1;
  } else if (ustawienia[ 5 + 14*device ] & 0x40) {
   on=0;
  }

if (on) {
   PORTD |= 0x04 << device;
  } else {
   PORTD &= ~(0x04 << device);
  }
}
#define alarm_ON() PORTD |= 0x10
#define alarm_OFF() PORTD &= 0xef
// z jakis powodow nie chial dzialac na PB7
//#define alarm_ON() PORTB |= 0x80
```

```
//#define alarm_OFF() PORTB &= 0x7f
//#define alarm_ON() asm volatile( "sbi 0x18,7" )
//#define alarm_OFF() asm volatile( "cbi 0x18,7" )


/** funkcja główna **/

int main() {
 opoznienie (10240);
 DDRB = 0xbf;
 DDRD = 0x1c;
 PORTD = 0x0;
 alarm[0] = 0;
 alarm[1] = 0;

/** odczyt konfiguracji **/
for (j=0; j<28; j++) {
  ustawienia[j] = eeprom_read_byte(j);
  eeprom_busy_wait();
 }

/** wyświetlacz - inicjalizacja **/
opoznienie(1000000);
 lcd_send_comand2 (0x2); lcd_send_comand (0x28);
  // Function Set (2 linie 4 bit) = 0010, 0010 10**
 lcd_send_comand (0xc);
  // On-Off Control (bez pokazywania kursora) = 0000 1100
 lcd_send_comand (0x6);
  // Entry Mode Set = 0000 0110

lcd_first_line();
 lcd_send_string ("   AKWARIUM  TERRARIUM  ", 24); // jeden wiersz po 24 znaki


/** port szeregowy - inicjalizacja **/
UCR = 1 << TXEN | 1 << RXEN | 1 << RXCIE;
  // włączenie nadajnika, odbionika UART wraz z generowaniem przerwań
 UBRR = 47; // szybkosc transmisji 9600 bps
 uart_set = -1;
 sti(); // włączenie przerwań


/** pętla główna **/
while(1) {
  /** termometry **/
  for (dn=0; dn<2; dn++) {
   // docelowo w glownej i=0,1 termometr + i*8 temperatura_ascii + i*5
   opoznienie(1000000);
   onewire_command(0x44, NULL);

   while ( !onewire_bit(1) );
   send_jeden();

   opoznienie(1000000);
   onewire_command(0xbe, ustawienia + 6 + 14*dn);
   t0 = onewire_reply();
   t1 = onewire_reply();

   temp_old = ustawienia[ 14*dn ];
   temp = ustawienia[ 14*dn ] = t1 << 4 | t0 >> 4;
```

```
    if (temp - temp_old > 10 || temp_old - temp > 10) continue;

    ustawienia[ 5 + 14*dn ] &= 0xf0;
     // wymazuje po przecinkowa czes temperatury (4 mlodsze bity)
    ustawienia[ 5 + 14*dn ] |= t0 & 0x0f;
     // i zapisuje ponownie

    /** generowanie napisu ASCII **/
    temperatura_ascii[ 4 + 10*dn ]='0'; temperatura_ascii[ 5 + 10*dn ]='0';
    itoa( temp , temperatura_ascii + 4 + 10*dn, 10);
    temperatura_ascii[ 6 + 10*dn ]='.';
    itoa( (t0 & 0x0f) * 625 , temperatura_ascii + 7 + 10*dn, 10);
    temperatura_ascii[ 8 + 10*dn ]=' ';

    /** podejmowanie decyzji i dzialan **/
    if ( temp < ustawienia[3 + 14*dn] ) {
     // alarm - za zimno
     temperatura_ascii[ 9 + 10*dn ]=185;
     grzanie(dn, 1);
     alarm[dn]++;
    } else if ( temp < ustawienia[1 + 14*dn] ) {
     // zaczynamy grzejemy
     temperatura_ascii[ 9 + 10*dn ]=27;
     grzanie(dn, 1);
     alarm[dn]=0;
    } else if ( temp < ustawienia[2 + 14*dn] ) {
     // temperatura dobra - nie zmieniamy stanu grzania
     temperatura_ascii[ 9 + 10*dn ]=205;
     alarm[dn]=0;
    } else if ( temp < ustawienia[4 + 14*dn] ) {
     // przestajemy grzac
     temperatura_ascii[ 9 + 10*dn ]=26;
     grzanie(dn, 0);
     alarm[dn]=0;
    } else {
     // alarm - za goraco
     temperatura_ascii[ 9 + 10*dn ]=186;
     grzanie(dn, 0);
     alarm[dn]++;
    }
   }

   /** obsługa alarmu **/
   if (alarm[0] == 0 && alarm[1] == 0) {
    alarm_OFF();
   } else if (alarm[0] > 0 || alarm[1] > 0) {
    alarm_ON();
   }

   /** obsługa LCD **/
   for (dn=0; dn<4; dn++) temperatura_ascii[dn] = ' ';
   for (dn=10; dn<14; dn++) temperatura_ascii[dn] = ' ';
   for (dn=20; dn<24; dn++) temperatura_ascii[dn] = ' ';
   lcd_second_line();
   lcd_send_string (temperatura_ascii, 24);
  }
 }
```

ONEWIRE_GET_ROM.C

```
/**

 * program do odczytu numeru ROM urządzenia OneWire
 * w tym celu pojedyńcze urządzenie podłączyć do komputera PC
 * (w obecnej konfiguracji linia danych do pinu 16 portu LPT,
 * należy też podłączyć masę, zasilania nie trzeba podłączać)
**/


/*   */

#include <stdio.h>
#include <sys/io.h>

#define bool char

#define cli() asm("cli")
#define sti() asm("sti")


long long czas() { asm( "rdtsc" ); }
void _delay_us( long long l ) {
 l = l * 3012 + czas(); // prędkość procesora w MHz
 while( l > czas() );
}
#define _delay_ms(a) _delay_us( 1000*(a) )


#define onewire_send_zero()   outb(0x00, 0x37a)
#define onewire_send_jeden()  outb(0x04, 0x37a)
#define onewire_read()        (inb(0x37a) & 0x04) >> 2

#include "onewire.c"

main(){
 iopl (3);

unsigned char i, c;
 onewire_reset();
 onewire_byte(0x33);

printf("const char rom[8] = {");
 for ( i = 0; i < 8 ; ++i ) {
  c = onewire_reply();
  printf("0x%02x", c);
  if (i<7) printf(", ");
 }
 printf("};\n");
}
```

Sterownik LCD z komunikacją RS232 w VHDL

Niniejszy kod ten powstał w ramach zaliczenia przedmiotu Projektowanie Programowalnych Układów Scalonych na Wydziale Elektroniki i Technik Informacyjnyh Politechniki Warszawskiej. Zadanie polegało na stworzeniu układu sterującego alfanumerycznym wyświetlaczem LCD opartym o sterownik zgodny z HD44780[2194] w postaci

oprogramowania układu FPGA. Układ miał odbierać dane i komendy sterujące z wykorzystaniem portu szeregowe UART.

Projekt został zrealizowany w postaci kodu VHDL. Posiada konstrukcję modularną, wydzielić można następujące moduły:

- układ odbierający dane z postu szeregowego (uart_rxd.vhdl)
- układ zapewniający komunikację z wyświetlaczem LCD - wysyłanie komend i danych (lcd.vhdl)
- układ sterujący (sterownik.vhdl)
- połączenia pomiędzy modułami zostały opisane w polaczenie.vhdl

Wszystkie trzy układy sterujące zostały zrealizowane w formie automatów.

Odbiór danych z UART

Jest to bardzo prosta implementacja odbiornika UART zrealizowana w formie automatu o 11 stanach. Stany numerowane są od 0 do 10, wyróżnić można następujące grupy stanów:

- stan 0 - idle, automat oczekuje na bit startu
- stan 1 - automat oczekuje potwierdzenia że jest to bit startu (możliwe przejście do stanu 2 lub 0)
- stany 2-9 - odpowiadają odbiorowi kolejnych bitów
- stan 10 - automat sprawdza czy jest bit stopu, jeżeli tak wystawia informacje że odebrał bajt (sygnał `data_ready`) oraz odebrany bajt (szyna `data_out`)

Za wyjątkiem stanu 1 możliwe jest przejście tylko do stanu następnego (dla stanu 10 stanem następnym jest 0). Przejścia do kolejnych stanów odbywają się w momencie gdy licznik `RX_CLK_CNT` osiągnie połowę wartości maksymalnej (`RX_CLK_DIV - 1`) - wtedy jest wystawiany sygnał zezwolenia dla układu czytającego dane i wybierającego stan następny. Wyjątkiem jest przejście ze stanu 0 do stanu 1 które odbywa się na skutek detekcji zera na linii `rxd`, z przejściem tym związane jest także wyzerowanie i wystartowanie licznika `RX_CLK_CNT`.

UART_RXD.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity UART_RXD is
 generic (
  RX_CLK_DIV: natural := 104166
 );
 port (
  rxd: in std_logic;
  clk, reset: in std_logic;
  data_out: out std_logic_vector (7 downto 0);
  data_ready: out std_logic
 );
end;

architecture logic of UART_RXD is
 signal RX_CLK_CNT : integer;
 signal RX_CLK_EN : std_logic;
 signal RX_BUFF : std_logic_vector (7 downto 0);
 signal RS_STATE, RS_STATE_NEXT : integer range 0 to 10;
begin
 process (reset, clk)
 begin
  -- obsluga resetu
  if (reset = '1') then
   RS_STATE <= 0;
   RX_CLK_EN <= '0';
  elsif ( rising_edge(clk) ) then
   -- obsluga poczatku bitu startu (start licznika)
```

```
      if (RS_STATE = 0) then
        if (rxd = '0') then
          -- rozpoczynamy odliczanie wewnetrznego zegara
          RX_CLK_CNT <= 0;
          -- przechodzimy do stanu start (potwierdzanie odebrania bitu startu)
          RS_STATE <= 1;
        end if;
      -- obsluga licznika
      else
        if (RX_CLK_CNT = RX_CLK_DIV - 1) then
          -- doliczenie do wartosci dzielnika zegara powoduje:
          --    wyzerowanie licznika
          RX_CLK_CNT <= 0;
          --    przejscie do nowego stanu
          RS_STATE <= RS_STATE_NEXT;
        else
          RX_CLK_CNT <= RX_CLK_CNT + 1;
        end if;

        if (RX_CLK_CNT = (RX_CLK_DIV - 1)/2) then
          -- doliczenie do polowy wartosci dzielnika zegara powoduje:
          --    wystawienie przez jeden takt zegara sygnalu RX_CLK_EN
          RX_CLK_EN <= '1';
        else
          RX_CLK_EN <= '0';
        end if;
      end if;
    end if;
  end process;

process (clk)
  begin
    if ( falling_edge(clk) ) then
      if( RX_CLK_EN = '1' ) then
        case RS_STATE is
          when 1 => -- bit startu
            if (rxd = '0') then
              data_ready <= '0';
              RS_STATE_NEXT <= 2;
            else
              -- blad (nie ma bitu startu) - wracamy do IDLE
              RS_STATE_NEXT <= 0;
            end if;
          when 10 => -- bit stopu
            RS_STATE_NEXT <= 0;
            if (rxd = '1') then
              -- skompletowalismy bajt
              data_out <= RX_BUFF;
              data_ready <= '1';
            end if;
          when 0 =>
            RS_STATE_NEXT <= 0;
          when others =>
            RX_BUFF(RS_STATE-2) <= rxd;
            RS_STATE_NEXT <= RS_STATE_NEXT + 1;
        end case;
      end if;
    end if;
  end process;
```

```
end;
```

Sterowanie wyświetlaczem LCD

Układ sterujący komunikacją z LCD ma zachowanie odpowiednich zależności czasowych w komunikacji na szynie LCD. Zrealizowany jest jako 5 stanowy automat o następujących stanach:

- IDLE - oczekiwanie na sygnał ważności danych na szynie wejściowej
- START - początek transmisji - ustawienie sygnału wybierającego pomiędzy trybem rozkazowym a danych
- E_UP - wystawienie danych na szynę i podniesienie strobu
- E_DOWN - opuszczenie strobu
- WAIT_TIME - stan oczekiwania

Stany przełączane są kolejno (IDLE - START - E_UP - E_DOWN - IDLE), z tym że przejścia (za wyjątkiem przejść do oraz z IDLE) odbywają się przez stan WAIT_TIME. Stan ten korzysta z dodatkowych informacji (ustawianych przed wejściem do tego stanu) - stanu do którego ma automat przejść po zakończeniu oczekiwania (`LCD_STATE_AFTER_WAIT`) oraz liczby cykli zegara przez które należy czekać (`LCD_CLK_DIV`).

LCD.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity LCD_DRV is
 generic (
  LCD_CLK_CNT_40ns: natural := 40;
  LCD_CLK_CNT_230ns: natural := 230
 );
 port (
  data_in: in std_logic_vector (7 downto 0);
  rs_mode: in std_logic;
  lcd_run: in std_logic;
  lcd_done: out std_logic;
  clk, reset: in std_logic;

  data_out: out std_logic_vector (7 downto 0);
  rs_out, e_out, rw_out: out std_logic
 );
end;


architecture logic of LCD_DRV is
 type STANY is (
  IDLE,
  WAIT_TIME,
  START,
  E_UP,
  E_DOWN
);

signal LCD_CLK_CNT, LCD_CLK_DIV : integer;
 signal LCD_STATE, LCD_STATE_NEXT, LCD_STATE_AFTER_WAIT : STANY;
 signal LCD_DATA : std_logic_vector (7 downto 0);
 signal LCD_RS : std_logic;
begin
 process (clk, reset)
 begin
  if ( reset = '1' ) then
```

```
    LCD_STATE <= IDLE;
  elsif ( falling_edge(clk) ) then
    if (LCD_STATE /= IDLE) then
      LCD_STATE <= LCD_STATE_NEXT;
    else
      if ( lcd_run = '1' ) then
        -- startujemy licznik zwiekszany co LCD_CLK_DIV tak aby rosl o 1 nie czesciej niz co 15
        ns
        LCD_STATE <= START;
        LCD_DATA <= data_in;
        LCD_RS <= rs_mode;
      end if;
    end if;
  end if;
end process;

process (clk) begin
  if( falling_edge(clk) ) then
    case LCD_STATE is
      when START =>
        lcd_done <= '0';
        rs_out <= LCD_RS;
        rw_out <= '0';

        LCD_STATE_AFTER_WAIT <= E_UP;
        LCD_CLK_DIV <= LCD_CLK_CNT_40ns;
        LCD_STATE_NEXT <= WAIT_TIME;

      when E_UP =>
        data_out <= LCD_DATA;
        e_out <= '1';

        LCD_STATE_AFTER_WAIT <= E_DOWN;
        LCD_CLK_DIV <= LCD_CLK_CNT_230ns;
        LCD_STATE_NEXT <= WAIT_TIME;

      when E_DOWN =>
        e_out <= '0';

        LCD_STATE_AFTER_WAIT <= IDLE;
        LCD_CLK_DIV <= LCD_CLK_CNT_230ns;
        LCD_STATE_NEXT <= WAIT_TIME;

      when IDLE =>
        e_out <= '0';
        lcd_done <= '1';
        LCD_CLK_CNT <= 0;

        LCD_STATE_NEXT <= START;

      when WAIT_TIME =>
        if (LCD_CLK_CNT = LCD_CLK_DIV) then
          LCD_CLK_CNT <= 0;
          LCD_STATE_NEXT <= LCD_STATE_AFTER_WAIT;
        else
          LCD_CLK_CNT <= LCD_CLK_CNT + 1;
        end if;
    end case;
  end if;
end process;
```

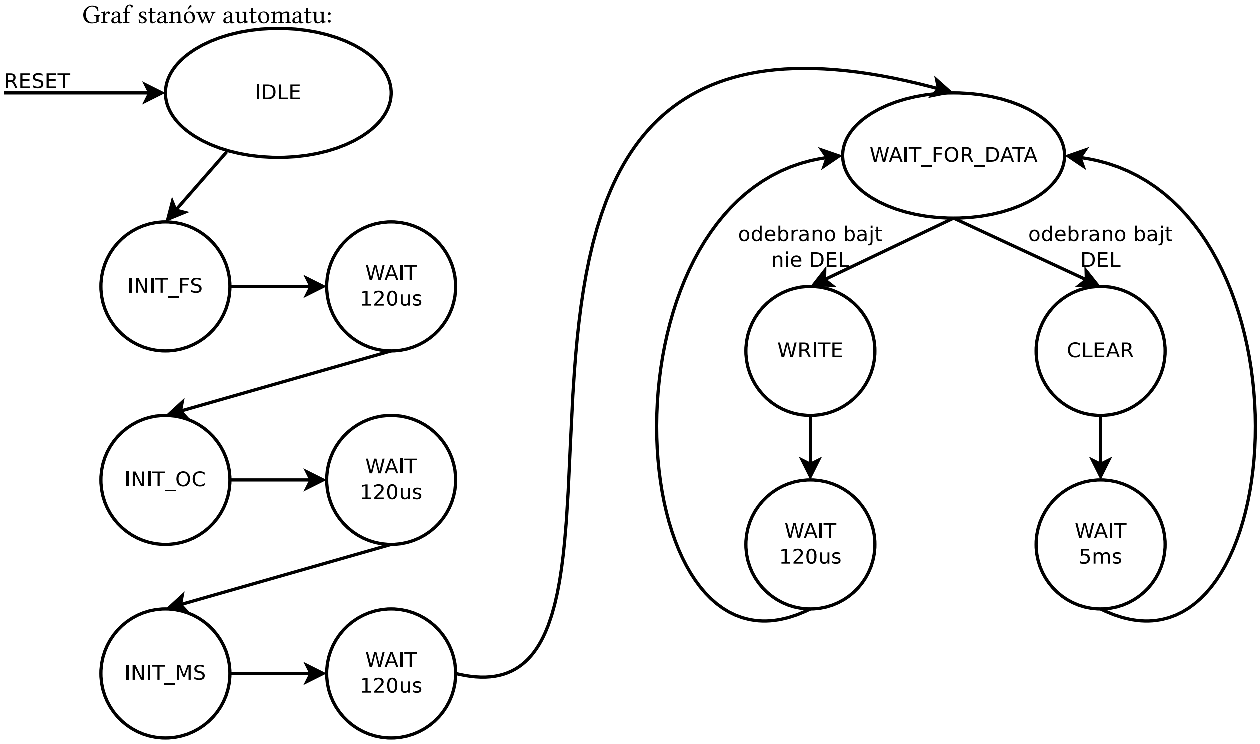
```
end;
```

Główny automat sterujący

Układ ten odpowiedzialny jest za inicjalizację wyświetlacza oraz wysyłanie do niego danych odebranych z UART. W przypadku gdy z UART zostanie odebrany znak DEL (kod 0x7f) wysyłana jest komenda czyszcząca wyświetlacz. Automat posiada następujące stany:

- IDLE - stan początkowy, przejście do niego na skutek sygnału reset
- INIT_FS - pierwszy krok inicjalizacji LCD - "Function Set"
- INIT_OC - drugi krok inicjalizacji LCD - "On-Off Control" - włączenie wyświetlacza i wyłączenie kursora
- INIT_MS - trzeci krok inicjalizacji LCD - "Mode Set" - automatyczne przesuwanie w prawo przy zapisie
- WAIT_FOR_DATA - Oczekiwanie na odbiór bajtu z UART
- CLEAR - Wyczyszczenie wyświetlacza LCD
- WRITE - Wysłanie odebranego bajtu do wyświetlacza LCD
- WAIT_TIME - Tryb oczekiwania przez `WAIT_CLK_DIV` taktów zegara, po nim przejście do stanu wskazanego przez `STAN_AFTER_WAIT`

Graf stanów automatu:



STEROWNIK.VHDL

```vhdl
library ieee;
use ieee.std_logic_1164.all;

entity STEROWNIK is
 generic (
  WAIT_CLK_DIV_120us: natural := 120000;
  WAIT_CLK_DIV_5ms : natural := 2880000000 -- 24000 * WAIT_CLK_DIV_120us
 );
 port (
  rxd_data: in std_logic_vector (7 downto 0);
  rxd_ready: in std_logic;
  clk, reset: in std_logic;

  lcd_data: out std_logic_vector (7 downto 0);
```

```
  lcd_rs: out std_logic;
  lcd_run: out std_logic
 );
end;

architecture logic of STEROWNIK is
 – constant WAIT_CLK_DIV_5ms : natural := 24000 * WAIT_CLK_DIV_120us;

type STANY is (
  IDLE,
  INIT_FS, – Function Set
  INIT_OC, – On-Off Control (display on, cursor off)
  INIT_MS, – Mode Set (move right)
  WAIT_FOR_DATA, – Wait for data from UART
  CLEAR, – Clear LCD
  WRITE, – Write Data to LCD
  WAIT_TIME
);

signal STAN, NEXT_STAN, STAN_AFTER_WAIT: STANY;
 signal WAIT_CLK_CNT, WAIT_CLK_DIV: integer;
 signal REC_DATA: std_logic_vector (7 downto 0);
begin
 process (clk, reset)
 begin
  if (reset='1') then
   STAN <= IDLE;
  elsif ( falling_edge(clk) ) then
   STAN <= NEXT_STAN;
  end if;
 end process;

process (clk)
 begin
  if ( falling_edge(clk) ) then
   case STAN is
    when WAIT_TIME =>
     if (WAIT_CLK_CNT = WAIT_CLK_DIV) then
      – doliczenie do wartosci dzielnika zegara powoduje:
      –   wyzerowanie licznika
      WAIT_CLK_CNT <= 0;

      NEXT_STAN <= STAN_AFTER_WAIT;
     else
      WAIT_CLK_CNT <= WAIT_CLK_CNT + 1;
     end if;
     lcd_run <= '0';
    when IDLE =>
     lcd_run <= '0';
     WAIT_CLK_CNT <= 0;
     NEXT_STAN <= INIT_FS;

    when WRITE =>
     lcd_data <= REC_DATA;
     lcd_rs <= '1';
     lcd_run <= '1';
     STAN_AFTER_WAIT <= WAIT_FOR_DATA;
     WAIT_CLK_DIV <= WAIT_CLK_DIV_120us;
     NEXT_STAN <= WAIT_TIME;
    when CLEAR =>
```

```
      lcd_data <= "00000001";
      lcd_rs <= '0';
      lcd_run <= '1';
      STAN_AFTER_WAIT <= WAIT_FOR_DATA;
      WAIT_CLK_DIV <= WAIT_CLK_DIV_5ms;
      NEXT_STAN <= WAIT_TIME;

     when INIT_FS =>
      lcd_data <= "001111-";
      lcd_rs <= '0';
      lcd_run <= '1';
      STAN_AFTER_WAIT <= INIT_OC;
      WAIT_CLK_DIV <= WAIT_CLK_DIV_120us;
      NEXT_STAN <= WAIT_TIME;
     when INIT_OC =>
      lcd_data <= "00001100";
      lcd_rs <= '0';
      lcd_run <= '1';
      STAN_AFTER_WAIT <= INIT_MS;
      WAIT_CLK_DIV <= WAIT_CLK_DIV_120us;
      NEXT_STAN <= WAIT_TIME;
     when INIT_MS =>
      lcd_data <= "00000110";
      lcd_rs <= '0';
      lcd_run <= '1';
      STAN_AFTER_WAIT <= WAIT_FOR_DATA;
      WAIT_CLK_DIV <= WAIT_CLK_DIV_120us;
      NEXT_STAN <= WAIT_TIME;

     when WAIT_FOR_DATA =>
      if ( rxd_ready = '1' ) then
       REC_DATA <= rxd_data;
       if (REC_DATA = "01111111") then
        NEXT_STAN <= CLEAR;
       else
        NEXT_STAN <= WRITE;
       end if;
      end if;
    end case;
   end if;
  end process;
end;
```

POLACZENIE.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity PRUS1 is
 port (
  rxd: in std_logic;
  clk, reset: in std_logic;
  lcd_BUS: out std_logic_vector (10 downto 0)
 );
end;

architecture logic of PRUS1 is
 signal uart_DATA, lcd_DATA: std_logic_vector (7 downto 0);
```

```
 signal uart_REDY, lcd_RS, lcd_RUN: std_logic;
begin
 uart: entity work.UART_RXD(logic)
  generic map(RX_CLK_DIV => 52083) -- 1/9600 s
  port map(
   clk => clk,
   reset => reset,

   rxd => rxd,

   data_out => uart_DATA,
   data_ready => uart_REDY
  );
 ster: entity work.STEROWNIK(logic)
  generic map(
   WAIT_CLK_DIV_120us => 60000, -- 24
   WAIT_CLK_DIV_5ms => 2500000
  )
  port map(
   clk => clk,
   reset => reset,

   rxd_data => uart_DATA,
   rxd_ready => uart_REDY,

   lcd_data => lcd_DATA,
   lcd_rs => lcd_RS,
   lcd_run => lcd_RUN
  );
 lcd: entity work.LCD_DRV(logic)
  generic map(
   LCD_CLK_CNT_40ns => 20,
   LCD_CLK_CNT_230ns => 165
  )
  port map(
   clk => clk,
   reset => reset,

   data_in => lcd_DATA,
   rs_mode => lcd_RS,
   lcd_run => lcd_RUN,

   data_out => lcd_BUS (10 downto 3),
   rs_out => lcd_BUS (0),
   e_out => lcd_BUS (1),
   rw_out => lcd_BUS (2)
  );
end;
```

TESTBENCH.VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity testbench is
end;

architecture logic of testbench is
```

```
 signal CLK, RST: std_logic := '1';

signal RXD, RXD_RUN: std_logic := '0';
 signal RXD_CNT: integer := 0;
 signal RXD_IN: std_logic_vector (7 downto 0) := "01010011";
begin
 – zegar
 process
  begin
  wait for 1 ns;
  CLK <= not CLK;
 end process;

– zatrzymanie symulacji
 process
  begin
  wait for 1200 us;
  assert FALSE
   report "koniec symulacji"
  severity FAILURE;
 end process;

RST <= '1', '0' after 0.5 ns;

–RXD_IN <= "01011010" after 162 ns;
 –RXD_RUN <= '1' after 82 ns, '0' after 162 ns, '1' after 170 ns;
 RXD_RUN <= '1' after 20 ns;
 process
  begin
   if (RXD_RUN = '1') then
    if (RXD_CNT = 0) then
     RXD <= '0'; – bit startu
     RXD_CNT <= 1;
    elsif (RXD_CNT < 9) then
     RXD <= RXD_IN(RXD_CNT-1);
     RXD_CNT <= RXD_CNT + 1;
    else
     RXD <= '1'; – bit stopu
    end if;
   else
    RXD <= '1'; – cisza na linii
    RXD_CNT <= 0;
   end if;
   wait for 104166 ns;
 end process;

calosc: entity work.PRUS1(logic)
  port map(
   clk => CLK,
   reset => RST,
   rxd => RXD
  );
end;
```

CommonsConfig-base.sh

```
#!/bin/bash
```

```
# włączenie konfiguracji
. $HOME/.CommonsConfig.conf

switch_bzr() {
 if [ -L .bzr ]; then
  /bin/rm .bzr
elif [ -e .bzr ]; then
  print_error "`pwd`/.bzr exists but is not a symbolic link"
  exit
 fi
 ln -s .CommonsConfig-BZR-$1 .bzr
}

show_maanged_files() {
 for repo in $BZR_REPO_LIST; do
  echo "BZR - $repo:"
  switch_bzr "$repo"
  bzr ls -r last:1 | while read l; do echo "  $1"; done
 done
 echo "SSHFS:"
echo "$SSHFS_LINK_LIST" | while read r l; do echo "  $1 (as $r)"; done
}



# funkcja wypisująca komunikat o błedzie (na konsolę i poprzez kdialog
print_error() {
 echo -e "\\033[1;31mERROR: $1\\033[0m"
kdialog --error "$1"
}

print_timeout_error() {
 print_error "$2" &
#   sleep $1
#   kill %1
}

progress_bar_run() {
 dcopserver
progress_bar_info="$1"
progress_bar_progress=0
 progress_bar=`kdialog --progressbar \
  "$progress_bar_info:                                                  " $2`
progress_bar=`dcopclient "$progress_bar"`
}
progress_bar_update() {
 progress_bar_progress=$(( $progress_bar_progress + 1 ))
 dcop $progress_bar ProgressDialog setProgress $progress_bar_progress
 dcop $progress_bar ProgressDialog setLabel "$progress_bar_info: $1"
}
progress_bar_close() {
 dcop $progress_bar ProgressDialog close
}



start_CommonsConfig() {
 progress_bar_run "starting commons config system" $(( `echo $BZR_REPO_LIST | wc -w` + 3 ))
 cd
```

```
echo '*' > .bzrignore

# aktualizacja danych z repozytoriów BZR
 for repo in $BZR_REPO_LIST; do
  progress_bar_update "upgrade BZR \"$repo\" repo"
  switch_bzr "$repo"
  bzr pull "sftp://$BZR_HOST/$BZR_REMOTE_DIR/$repo"
  bzr up
done

progress_bar_update "mounting SSHFS resources"

# sprawdzamy czy zamontowany SSHFS
 if mount | grep .CommonsConfig-SSHFS > /dev/null; then
   echo "SSH-FS is mounted"
else
  # montowanie SSHFS
  if [ -L .CommonsConfig-SSHFS ]; then
   /bin/rm .CommonsConfig-SSHFS
  fi
  mkdir -p .CommonsConfig-SSHFS
  sshfs -o workaround=rename "$SSHFS_HOST:$SSHFS_REMOTE_DIR" .CommonsConfig-SSHFS

  # jeżeli montowanie sshfs się nie udało to korzystamy z kopii lokalnej
  if [ $? != 0 ]; then
   print_error "Unable to mount $SSHFS_HOST:$SSHFS_REMOTE_DIR via sshfs. Use bzr local ⤸
   reposity."

   # zwolnienie katalogu
   if ! rmdir .CommonsConfig-SSHFS; then
    mv .CommonsConfig-SSHFS .CommonsConfig-SSHFS.error_backup_`date +"%s"`
   fi

   # utwotrzenie dowiązania
   ln -s .CommonsConfig-SSHFS.local_backup .CommonsConfig-SSHFS
  fi
 fi

# tworzenie dowiązań do danych na SSHFS
 progress_bar_update "prepare link to SSHFS resources"
echo "$SSHFS_LINK_LIST" | while read remote local; do
  if [ -L "$local" ]; then
   /bin/rm "$local"
  elif [ -e "$local" ]; then
   print_error "$local is not symlink"
   if [ -e ~/.CommonsConfig-SSHFS/$remote ]; then
    mv "$local" "$HOME/.CommonsConfig-SSHFS/$remote.`date +"%s"`"
    mv "$local" "$local.`date +"%s"`"
   else
    cp -r "$HOME/$local" "$HOME/.CommonsConfig-SSHFS/$remote" &&
     mv "$local" "$local.`date +"%s"`"
   fi
  fi
  ln -s "$HOME/.CommonsConfig-SSHFS/$remote" "$HOME/$local"
done

progress_bar_close
}

stop_CommonsConfig() {
```

```
 progress_bar_run "stoping commons config system" $(( `echo $BZR_REPO_LIST_AUTOPUSH | wc -w` +
 2 ))

progress_bar_update "updating local copy of SSHFS"
rsync -rltu ~/.CommonsConfig-SSHFS/ ~/.CommonsConfig-SSHFS.local_backup/

# aktualizacja danych z repozytoriów BZR
 for repo in $BZR_REPO_LIST_AUTOPUSH; do
  progress_bar_update "pushing BZR \"$repo\" repo"
  switch_bzr "$repo"
  bzr_auto_add "$repo"
  bzr ci -m "automatyczna auktaulizacja" > /tmp/$PPID.bzr-$repo-ci.stdout 2>/tmp/$PPID.bzr-
  $repo-ci.stderr
  if [ $? -ne 0 -a $? -ne 3 ]; then #FIXME sprawdzic kod powr przy prawdziwych błędach
   error=`cat /tmp/$PPID.bzr-$repo-ci.stderr`
   print_timeout_error 30 "push_bzr: $error"
  fi

  bzr push "sftp://$BZR_HOST/$BZR_REMOTE_DIR/$repo" > /tmp/$PPID.bzr-$repo-push.stdout 2>
  /tmp/$PPID.bzr-$repo-push.stderr
  if [ $? -ne 0 ]; then
   error=`cat /tmp/$PPID.bzr-$repo-push.stderr`
   print_timeout_error 30 "push_bzr: $error"
  fi
 done

progress_bar_close
}

# funkcja instalująca system
install_mount_script() {
 # instalacja skryptu montującego dla KDE
 # możnaby także dopisać w ~/.bash_profile
 #      if [ "$DISPLAY" != "" ]; then MOUNT_CMD fi
 mkdir -p $HOME/.kde/env
echo "#!/bin/bash

 $HOME/.CommonsConfig-base.sh start
 " > $HOME/.kde/env/commons-config-mount.sh
chmod +x $HOME/.kde/env/commons-config-mount.sh

# instalacja skryptu odmontowywującego dla KDE
 # możnaby także dopisać w ~/.bash_logout
 #      if [ "$DISPLAY" != "" ]; then UMOUNT_CMD fi
 # ale KDE nie uwzględnia tego pliku
 mkdir -p $HOME/.kde/shutdown
echo "#!/bin/bash

 $HOME/.CommonsConfig-base.sh stop
 " > $HOME/.kde/shutdown/commons-config-umount.sh
chmod +x $HOME/.kde/shutdown/commons-config-umount.sh
}

case $1 in
 "start")
  start_CommonsConfig
;;
 "stop")
  stop_CommonsConfig
;;
```

esac

CommonsConfig.conf

```
# lista repozytoriów automatycznie aktualizowanych przy zamykaniu systemu
#  nazwy wymienione na liście wchodzą w skład nazwy katalogu .bzr danego repo w ⤸
nastepujacy sposob:
#  .CommonsConfig-BZR-nazwa
#  jako że nazwy są powiązane z ścieżką składowania kopii na serwerze nie jest ⤸
możliwe używanie takiejsamej nazwy
#  na dwuch hostach jeżeli chcemy aby były to różne repozytoria
BZR_REPO_LIST_AUTOPUSH="global `hostname`"

# lista repozytoriów bzr których kopie robocze aktualizujemy przy starcie systemu
BZR_REPO_LIST="$BZR_REPO_LIST_AUTOPUSH "

# lista plików przechowywanyach na SSHFS złożona z linii postaci:
#  sciezka/do_pliku/na_sshf sciezka/do_pliku/w_home
SSHFS_LINK_LIST="psi-history .psi/profiles/domyslny/history
knotes.ics .kde/share/apps/knotes/notes.ics
addressbook-dir .kde/share/apps/kabc/stdvcf"

# lista plików trzymanych w bzr
bzr_auto_add() {
case $1 in
"global")
  ADD_LIST=".kde/share/apps/konqueror/bookmarks.xml
  .kde/share/apps/kabc/std.vcf
  .kde/share/config/kmailrc
  .kde/share/config/emaildefaults
  .kde/share/config/emailidentities
  .kde/share/config/knoderc
  .kde/share/apps/knode
  .linphonerc
  .kde/share/services/searchproviders
  .kde/share/config/konversationrc
  .gEDA
  .pcb
  .tex
  .w3m/config
  .bashrc
  .ssh/config
  .vimrc
  .gnupg
  .CommonsConfig-SSHFS.local_backup"
;;
"`hostname`")
  ADD_LIST="
  .ssh/authorized_keys
  .psi/profiles/domyslny/config.xml
  .psi/profiles/domyslny/options.xml
  .bin"
;;
esac
  bzr add $ADD_LIST
}

# konfiguracja dostępu do hosta udostępniającego BZR
```

```
BZR_HOST=""
BZR_REMOTE_DIR=""

# konfiguracja dostępu do hosta udostępniającego SSHFS
SSHFS_HOST="$BZR_HOST"
SSHFS_REMOTE_DIR="$BZR_REMOTE_DIR/SSHFS"
```

CommonsConfig-addionals.sh

```
#!/bin/bash

# funkcja wypisująca komunikat w okienkach dialogowych KDE
# dostosowuje typ komunikatu do jego wielkości
#    (czy sam komunikat z z dodatkowym info z podanych plików)
# wymaga przekierowanie wyjść do plików poptrzez
# exec 2> /tmp/log_stderr.${PPID}pid 1> /tmp/log_stdout.${PPID}pid
# lub przekierowania w komendach
print_msg() {
 if [ -s "/tmp/log_stderr.${PPID}pid" -o -s "/tmp/log_stdout.${PPID}pid" ]; then
  kdialog --textinputbox "$1:" \
   "`cat /tmp/log_stderr.${PPID}pid; echo -e "\n==============\n"; cat /tmp/log_stdout.${PPID}⤸
   pid`"
else
  kdialog --msgbox "$1"
fi
 :> /tmp/log_stderr.${PPID}pid
:> /tmp/log_stdout.${PPID}pid
}

# funkcja pakująca linki z katalogu wraz z pakowaniem ich celów
#    gdy cel jest w $HOME to pomijamy $HOME w ścieżce
link_dir_backup() {
 pushd $HOME > /dev/null
# pakowanie linków
 tar -cf /tmp/konfiguracja.${PPID}pid.tar `echo "$1" | grep ^$HOME | sed "s%^$HOME/*%%g"`
# lista celów linków
 FULL_LIST=`find "$1" -type l | while read f; do readlink -f "$f"; done`
# pakujemy cele linków - jeżeli w $HOME to bez początkowej ścieżki (musi być ⤸
 uruchamiane w $HOME)
 tar -rf /tmp/konfiguracja.${PPID}pid.tar \
  `echo "$FULL_LIST" | grep ^$HOME | sed "s%^$HOME/*%%g"`\
  `echo "$FULL_LIST" | grep -v ^$HOME`
# kopiuuemy + komounikat
 move_backup /tmp/links.${PPID}pid.tar "$2" "tar"
popd > /dev/null
}

###############################################
##    FUNKCJE EKSPORTU DO PRZENOŚNEJ POSTACI    ##
###############################################

# funkcja eksportująca klucze GnuPG
gpg_export() {
 gpg --list-keys | awk '
  $1=="pub" {
   split($2, tmp, "/"); ctrl=tmp[2];
  }
  $1=="uid" && ctrl!=0 {
```

```
   tmp2=""
   for(i=2; i<=NF; i++)
    tmp2=sprintf("%s %s",tmp2, $i)
   printf("%s %s\n" , tmp2, ctrl)
   ctrl=0
  }
 ' | while read key_name; do
  key_id=${key_name/*</}
  key_id=${key_id/>*/}
  key_file=${key_name// /_}
  key_file=${key_file/<*>/}
  key_file=`echo $key_file | tr '/' '_'`".asc"

  gpg -a --export $key_id > "/tmp/key_exp.${PPID}pid"
  move_backup "/tmp/key_exp.${PPID}pid" "$1/$key_file" "asc"
done
}

# funkcja exportująca certyfikaty S-MIME
x509_export() {
 # zaimportowane i wymienione w trustlist.txt muszą być także CA dołączone do systemu (⤦
 np. z przeglądarkami WWW)
 # musza posiadać relax (dalej nie są sprawdzane), podobnie jak nieautoryzowanych CA (np. ⤦
 własny)
 # dla oficjalnych CA dodałem dodatkową flagę po relax - CA
 egrep -v 'relax[\t ]+CA' ~/.gnupg/trustlist.txt |cut -f1 -d' ' | while read key_finger; do
  key_file=`gpgsm -a -k $key_finger | grep 'Subject:' | awk 'BEGIN {RS="/"; FS="="} $1=="CN" {⤦
  print $2}'`
  key_file="$key_file ${key_finger//:/}.pem"

  gpgsm -a --export $key_finger > "/tmp/key_exp.${PPID}pid"
  move_backup "/tmp/key_exp.${PPID}pid" "$1/$key_file" "pem"
done
}

# funkcja tworząca linki o nazwach opartych na polu FN w katalogu zadanym przez $1 do ⤦
plików v-card w katalogu zadanym przez $2
vcf_export() {
 rm -f "$1"/*
 pushd "$1" > /dev/null
for f in "$2"/*; do
  name="`grep '^FN:' "$f" | cut -f 2- -d':' | tr ' ' '_' | tr -d '\n\r'`.vcf"
  echo "$f ->> $name"
  if [ "$name" != "" -a ! -e "$name" ]; then
   rsync "$f" "$name"
  else
   rsync "$f" .
  fi
 done
 popd > /dev/null
}

# funkcja eksportu do ludzkiej postaci (klucze szyfrujące)
prepare_userfriendly_dir() {
 rm -f $1/PrivateArchive/Cryptography/pgp/*
 gpg_export "$1/PrivateArchive/Cryptography/pgp"

rm -f $1/PrivateArchive/Cryptography/x509/*
 x509_export "$1/PrivateArchive/Cryptography/x509"
```

```
rsync "$HOME/.ssh/known_hosts" "$1/PrivateArchive/Cryptography/ssh/known_hosts"

vcf_export "$1/PrivateArchive/AddressBook" "$SSHFS_MOUNT_DIR/configs/addressbook-dir"

print_msg "Eksport OpenPGP, X509, SSH i AddressBook zakończony"
}
```

Inne

Licencje

własność publiczna (public domain)

Jest najbardziej liberalną licencją (aż do tego stopnia że zasadniczo jej nie ma ...), stosowaną do wszelkiego typu prac. Objęcie nią danego dzieła oznacza że twórca przekazuje wszelkie prawa autorskie ogółu ludzkości - każdy może zrobić z tym co tylko zechce włącznie z przywłaszczeniem sobie praw autorskich (do swojej kopi - oryginał dalej pozostanie własnością publiczną). Ważną cechą jest też że licencja ta może dotyczyć tylko wersji binarnych. Zobacz w Wikipedii: Public domain[2195].

licencje typu BSD (MIT/X11)

Są to krótkie, bardzo liberalne licencje pozwalające na modyfikacje i redystrybucję programu zarówno w źródłowej jak i binarnej, wymaga tylko wymienienia oryginalnego autora w redystrybuowanej kopii programu. Tego typu licencja jest także (oprócz wspomnianych w tytule BSD i MIT/X11) licencja Creative Commons - uznanie autorstwa (CC-BY). Licencje te są kompatybilne między sobą a także z dowolną mniej liberalną licencją (kod dystrybuowany na takiej licencji może być użyty (lub włączony do jakiegoś większego projektu) np. na warunkach licencji copyleft czy też komercyjnej. Warto zaznaczyć że pisząc tu o licencji BSD mam na myśli jej zmodyfikowaną wersję - bez klauzuli ogłoszeniowej. Zobacz w Wikipedii: Licencja BSD[2196], Licencja X11[2197].

Zbilżoną do tej grupy licencją jest Licencja Artystyczna[2198], jednak w przypadku nieudostępnienia kodu społeczności lub autorom orginału wymagana jest zmiana nazwy projektu oraz nazw plików wykonywalnych. Najczęściej jest ona stosowana duzalnie z GNU GPL.

licencje copyleft (w tym GNU)

Licencje copyleft są liberalnymi licencjami chroniącymi wolność objętego nimi dzieła poprzez zabranianie zmiany warunków licencyjnych. Najpopularniejszą z licencji tego typu jest GNU General Public License, natomiast bliźniacza licencja GNU Library General Public License zezwala na łączenie z nie wolnymi modułami (nie pozwala jednak na redystrybucję w sposób nie wolny oryginału). Problemy z wolnością ma natomiast GNU Free Documentation License (stosowanie sekcji niezmiennych). Tego typu licencja jest także licencja Creative Commons - uznanie autorstwa + na tych samych warunkach (CC-BY-SA). Dużym problemem licencji copyleft jest wzajemna niekomaptybilność (powodowana jest ona głównie koniecznością zachowania tej samej licencji, co uniemożliwia zastosowanie licencji analogicznej, częstym rozwiązaniem jest podwójne licencjonowanie. Zobacz w Wikipedii: Copyleft[2199], GNU General Public License[2200], GNU Free Documentation License[2201], Licencja Creative Commons[2202].

inne wolne licencje

Oprócz omówionych tutaj licencji istnieje dużo innych wolnych licencji, jednak każdą z nich można uznać za bliską którejś z grup tu przedstawionych.

licencje półwolne

Wspomnieć trzeba również o istnieniu licencji półwolnych - dozwolone jest modyfikacja i redystrybucja, jednak nie w celach zarobkowych. Ważnym przypadkiem są licencje Creative Commons z atrybutem użycie niekomercyjne (CC-BY-NC-SA).

LICENCJE NIEWOLNE - BEZPŁATNE

Spotkać możemy również się z licencjami pozwalającymi na darmowe rozpowszechnianie jednak nie umożliwiającymi dokonywania modyfikacji (np. CC-BY-ND). Niekiedy zabraniają także wykorzystywania w celach komercyjnych (np. CC-BY-NC-ND).

LICENCJE NIEWOLNE - PŁATNE

Doczynienia mamy także z licencjami całkowicie niewolnymi i niepozwalającymi na darmowe korzystanie z kopii i dalsze rozpowszechnianie - nazywanych licencjami oprogramowania komercyjnego.

PODSUMOWANIE

Więcej informacji na temat licencji, wolnego oprogramowania, … znaleźć można pod adresami:
`http://www.fsf.org/philosophy/categories.pl.html`
`http://www.fsf.org/philosophy/license-list.pl.html`
`http://www.fsf.org/philosophy/free-sw.pl.html`
`http://www.debian.org/intro/free.pl.html`
`http://www.debian.org/social_contract.pl.html`
w Wikipedii: Otwarte Oprogramowanie[2203], Wolne Oprogramowanie[2204], Comparison of free software licences[2205]

Standardy (RFC, …)

RFC[2206] są serią numerowanych dokumentów Internet Engineering Task Force określających zdecydowaną większość standardów internetowych (w tym tych podstawowych jak IP, TCP, UDP, ICMP, … oraz metod przesyłu IP poprzez różne sieci fizyczne). W RFC możemy często też znaleźć informacyjne odniesienia do innych otwartych standardów (nie specyfikowane bezpośrednio w RFC) - takich jak HTML i inne standardy W3C[2207] (W3C czyli World Wide Web Consortium jest organizacją odpowiedzialną za standardy takie jak: HTML, CSS, DOM, XML i jego odmiany - w szczególności XHTML, XSL, XPath, MathML, SCG, SMIL, XForms, VoiceXML, …). W RFC został też określony protokół XMPP będący podstawą Jabbera (jego rozszerzenia specyfikowane są w XEP[2208]).

Kilka istotnych elementów świata informatyki standaryzowana jest przez organizacje takie jak ANSI, ISO (m.in. C, C++, ASCII[2209], Unicode[2210], format zapisu czasu - ISO 8601[2211] …), IEEE (wiele interfejsów sprzętowych - Ethernet(IEEE 802.3[2212]), FireWire(IEEE 1394[2213]), port równoległy (IEEE 1284[2214]), JTAG[2215] (IEEE 1149.1), ale również istotne aspekty softwerowe - przedewszystkim POSIX[2216] (IEEE 1003)). Część aspektów standarzyzowana jest poprzez różne firmy czy też konsorcja (np. USB Consortium - USB[2217], The Open Group[2218] - przdewszystkim Single UNIX Specification[2219] czyli (do niedawna) konkurencyjny POSIX (standard SUSv3[2220])).

Poniżej przedstawiam listę ważniejszych protokołów wraz z numerami głównych dokumentów RFC je specyfikujących oraz ich statusem (STANDARD / DRAFT STANDARD / PROPOSED STANDARD / …). Przy statucie STANDARD podawany jest też numer stosownego dokumentu STD. Lista ta nie jest kompletną listą RFC mówiących o tych protokołach, dlatego też zachęcam również do korzystania z listy oficjalnych standardów internetowych[2221] oraz wyszukiwarki RFC[2222] na stronach rfc-editor. Z koleji na stronach ISOC PL[2223] można przeczytać trochę o RFC po polsku.

Internet Protocol (IP, ICMP, …):

- RFC0791 Internet Protocol. (STANDARD -> STD0005)
- RFC0792 Internet Control Message Protocol. (STANDARD -> STD0005)
- RFC0919 Broadcasting Internet Datagrams. (STANDARD -> STD0005)
- RFC0922 Broadcasting Internet datagrams in the presence of subnets. (STANDARD -> STD0005)
- RFC0950 Internet Standard Subnetting Procedure. (STANDARD -> STD0005)
- RFC1112 Host extensions for IP multicasting. (STANDARD -> STD0005)
- RFC1256 ICMP Router Discovery Messages. (PROPOSED STANDARD)

- RFC3376 Internet Group Management Protocol, Version 3 (PROPOSED STANDARD)

IPv6:

- RFC2460 Internet Protocol, Version 6 (IPv6) Specification. (DRAFT STANDARD)
- RFC2462 IPv6 Stateless Address Autoconfiguration. (DRAFT STANDARD)
- RFC2463 Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. (DRAFT STANDARD)
- RFC3513 Internet Protocol Version 6 (IPv6) Addressing Architecture. (PROPOSED STANDARD)
- RFC2461 Neighbor Discovery for IP Version 6 (IPv6). (DRAFT STANDARD)
- RFC2466 Management Information Base for IP Version 6: ICMPv6 Group. (PROPOSED STANDARD)
- RFC3587 IPv6 Global Unicast Address Format. (INFORMATIONAL)

User Datagram Protocol (UDP):

- RFC0768 User Datagram Protocol. (STANDARD -> STD0006)

Transmission Control Protocol (TCP):

- RFC0793 Transmission Control Protocol. (STANDARD -> STD0007)

Domain Name System (DNS):

- RFC1034 Domain names - concepts and facilities. (STANDARD -> STD0013)
- RFC1035 Domain names - implementation and specification. (STANDARD -> STD0013)
- RFC3596 DNS Extensions to Support IP Version 6. (DRAFT STANDARD)

Simple Mail Transfer Protocol (SMTP):

- RFC0821 Simple Mail Transfer Protocol. (STANDARD -> STD0010)
- RFC1869 SMTP Service Extensions. (STANDARD -> STD0010)
- RFC0974 Mail routing and the domain system. (HISTORIC -> STD0010)
- RFC2920 SMTP Service Extension for Command Pipelining. (STANDARD -> STD0060)
- RFC2821 Simple Mail Transfer Protocol. (PROPOSED STANDARD)
- RFC1651 SMTP Service Extensions. (DRAFT STANDARD)
- RFC1652 SMTP Service Extension for 8bit-MIMEtransport. (DRAFT STANDARD)
- RFC1653 SMTP Service Extension for Message Size Declaration. (DRAFT STANDARD)
- RFC3461 Simple Mail Transfer Protocol (SMTP) Service Extension for Delivery Status Notifications (DSNs). (DRAFT STANDARD)
- RFC3463 Enhanced Mail System Status Codes. (DRAFT STANDARD)
- RFC2554 SMTP Service Extension for Authentication. (PROPOSED STANDARD)
- RFC3207 SMTP Service Extension for Secure SMTP over Transport Layer Security. (PROPOSED STANDARD)

Standard for the format of ARPA Internet text messages:

- RFC0822 Standard for the format of ARPA Internet text messages. (HISTORIC -> STD0011)
- RFC2822 Internet Message Format. (PROPOSED STANDARD)

Network News Transfer Protocol (NNTP):

- RFC0977 Network News Transfer Protocol. (PROPOSED STANDARD)

Telnet Protocol:

- RFC0854 Telnet Protocol Specification. (STANDARD -> STD0008)
- RFC0855 Telnet Option Specifications. (STANDARD -> STD0008)

File Transfer Protocol (FTP):

- RFC0959 File Transfer Protocol. (STANDARD -> STD0009)

Network Time Protocol (NTP):

- RFC1305 Network Time Protocol (Version 3) Specification, Implementation. (DRAFT STANDARD)

Hypertext Transfer Protocol:

- RFC2616 Hypertext Transfer Protocol – HTTP/1.1. (DRAFT STANDARD)
- RFC2617 HTTP Authentication: Basic and Digest Access Authentication. (DRAFT STANDARD)
- RFC2818 HTTP Over TLS. (INFORMATIONAL)

Multipurpose Internet Mail Extensions (MIME):

- RFC2045 Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. (DRAFT STANDARD)
- RFC2046 Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. (DRAFT STANDARD)
- RFC2047 MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text. (DRAFT STANDARD)
- RFC2049 Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples. (DRAFT STANDARD)

Post Office Protocol (POP):

- RFC1939 Post Office Protocol - Version 3. (STANDARD -> STD0053)

Internet Message Access Protocol (IMAP):

- RFC3501 INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1. (PROPOSED STANDARD)
- RFC3502 Internet Message Access Protocol (IMAP) - MULTIAPPEND Extension. (PROPOSED STANDARD)
- RFC3691 Internet Message Access Protocol (IMAP) UNSELECT command. (PROPOSED STANDARD)

Simple Network Management Protocol Version 3 (SNMPv3):

- RFC3411 An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. (STANDARD -> STD0062)
- RFC3412 Message Processing and Dispatching for the Simple Network Management Protocol (SNMP). (STANDARD -> STD0062)
- RFC3413 Simple Network Management Protocol (SNMP) Applications. (STANDARD -> STD0062)
- RFC3414 User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3). (STANDARD -> STD0062)
- RFC3415 View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP). (STANDARD -> STD0062)
- RFC3416 Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP). (STANDARD -> STD0062)
- RFC3417 Transport Mappings for the Simple Network Management Protocol (SNMP). (STANDARD -> STD0062)
- RFC3418 Management Information Base (MIB) for the Simple Network Management Protocol (SNMP). (STANDARD -> STD0062)
- RFC2819 Remote Network Monitoring Management Information Base. (STANDARD -> STD0059)
- RFC2572 Message Processing and Dispatching for the Simple Network Management Protocol (SNMP). (DRAFT STANDARD)

Secure Shell (SSH)

- RFC4250 The Secure Shell (SSH) Protocol Assigned Numbers (PROPOSED STANDARD)
- RFC4251 The Secure Shell (SSH) Protocol Architecture (PROPOSED STANDARD)
- RFC4252 The Secure Shell (SSH) Authentication Protocol (PROPOSED STANDARD)
- RFC4253 The Secure Shell (SSH) Transport Layer Protocol (PROPOSED STANDARD)
- RFC4254 The Secure Shell (SSH) Connection Protocol (PROPOSED STANDARD)
- RFC4255 Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints (PROPOSED STANDARD)
- RFC4256 Generic Message Exchange Authentication for the Secure Shell Protocol (SSH) (PROPOSED STANDARD)
- RFC4344 The Secure Shell (SSH) Transport Layer Encryption Modes (PROPOSED STANDARD)
- RFC4345 Improved Arcfour Modes for the Secure Shell (SSH) Transport Layer Protocol (PROPOSED STANDARD)

Transport Layer Security (TLS)

- RFC2246 The TLS Protocol Version 1.0. (PROPOSED STANDARD)
- RFC3546 Transport Layer Security (TLS) Extensions. (PROPOSED STANDARD)
- RFC3436 Transport Layer Security over Stream Control Transmission Protocol. (PROPOSED STANDARD)
- RFC2595 Using TLS with IMAP, POP3 and ACAP. (PROPOSED STANDARD)

Dynamic Host Configuration Protocol (DHCP):

- RFC2131 Dynamic Host Configuration Protocol. (DRAFT STANDARD)

Real-Time Protocol:

- RFC3550 RTP: A Transport Protocol for Real-Time Applications. (STANDARD -> STD0064)
- RFC3551 RTP Profile for Audio and Video Conferences with Minimal Control. (STANDARD -> STD0065)
- RFC2793 RTP Payload for Text Conversation. (PROPOSED STANDARD)

- RFC2833 RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals. (PROPOSED STANDARD)
- RFC2959 Real-Time Transport Protocol Management Information Base. (PROPOSED STANDARD)
- RFC2960 Stream Control Transmission Protocol. (PROPOSED STANDARD)

Session Description Protocol (SDP):

- RFC2327 SDP: Session Description Protocol. (PROPOSED STANDARD)
- RFC3264 An Offer/Answer Model with Session Description Protocol (SDP). (PROPOSED STANDARD)
- RFC3266 Support for IPv6 in Session Description Protocol (SDP). (PROPOSED STANDARD)
- RFC3388 Grouping of Media Lines in the Session Description Protocol (SDP). (PROPOSED STANDARD)
- RFC3407 Session Description Protocol (SDP) Simple Capability Declaration. (PROPOSED STANDARD)
- RFC3556 Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth. (PROPOSED STANDARD)
- RFC3605 Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP). (PROPOSED STANDARD)
- RFC3890 A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP). (PROPOSED STANDARD)

Session Initiation Protocol (SIP):

- RFC3261 SIP: Session Initiation Protocol. (PROPOSED STANDARD)
- RFC3262 Reliability of Provisional Responses in Session Initiation Protocol (SIP). (PROPOSED STANDARD)
- RFC3263 Session Initiation Protocol (SIP): Locating SIP Servers. (PROPOSED STANDARD)
- RFC3264 An Offer/Answer Model with Session Description Protocol (SDP). (PROPOSED STANDARD)
- RFC3265 Session Initiation Protocol (SIP)-Specific Event Notification. (PROPOSED STANDARD)
- RFC3515 The Session Initiation Protocol (SIP) Refer Method.
- RFC3891 The Session Initiation Protocol (SIP) "Replaces" Header. (PROPOSED STANDARD)
- RFC3893 Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format. (PROPOSED STANDARD)
- RFC3892 The Session Initiation Protocol (SIP) Referred-By Mechanism. (PROPOSED STANDARD)
- RFC3911 The Session Initiation Protocol (SIP) "Join" Header. (PROPOSED STANDARD)

Extensible Messaging and Presence Protocol (XMPP):

- RFC3920 Extensible Messaging and Presence Protocol (XMPP): Core. (PROPOSED STANDARD)
- RFC3921 Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence. (PROPOSED STANDARD)
- RFC3922 Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging (CPIM). (PROPOSED STANDARD)

Internet Relay Chat (IRC):

- RFC1459 Internet Relay Chat Protocol. (EXPERIMENTAL)
- RFC2813 Internet Relay Chat: Server Protocol. (INFORMATIONAL)
- RFC2812 Internet Relay Chat: Client Protocol. (INFORMATIONAL)
- RFC2811 Internet Relay Chat: Channel Management. (INFORMATIONAL)
- RFC2810 Internet Relay Chat: Architecture. (INFORMATIONAL)

Network File System (NFS):

- RFC3530 Network File System (NFS) version 4 Protocol. (PROPOSED STANDARD)

Lightweight Directory Access Protocol (LDAP):

- RFC2251 Lightweight Directory Access Protocol (v3). (PROPOSED STANDARD)

Voice Profile for Internet Mail:

- RFC3801 Voice Profile for Internet Mail - version 2 (VPIMv2). (DRAFT STANDARD)

Uniform Resource Identifier (URI): Generic Syntax:

- RFC3986 Uniform Resource Identifier (URI): Generic Syntax. (STANDARD -> STD0066)
- RFC1738 Uniform Resource Locators (URL). (PROPOSED STANDARD)

Unicode and ISO 8601:

- RFC3629 UTF-8, a transformation format of ISO 10646. (STANDARD -> STD0063)
- RFC2781 UTF-16, an encoding of ISO 10646. (INFORMATIONAL)
- RFC2152 UTF-7 A Mail-Safe Transformation Format of Unicode. (INFORMATIONAL)
- RFC3339 Date and Time on the Internet: Timestamps. (PROPOSED STANDARD)

XML, HTML, ...:

- RFC3076 Canonical XML Version 1.0. (INFORMATIONAL)
- RFC2854 The 'text/html' Media Type. (INFORMATIONAL)

Requirements for Internet Hosts and Router:

- RFC1122 Requirements for Internet Hosts - Communication Layers. (STANDARD -> STD0003)
- RFC1123 Requirements for Internet Hosts - Application and Support. (STANDARD -> STD0003)
- RFC1812 Requirements for IP Version 4 Routers. (PROPOSED STANDARD)

IP over Ethernet:

- RFC0826 Ethernet Address Resolution Protocol. (STANDARD -> STD0037)
- RFC0903 Reverse Address Resolution Protocol. (STANDARD -> STD0038)
- RFC0894 Standard for the transmission of IP datagrams over Ethernet networks. (STANDARD -> STD0041)
- RFC1042 Standard for the transmission of IP datagrams over IEEE 802 networks. (STANDARD -> STD0043)

The Point-to-Point Protocol (PPP):

- RFC1661 The Point-to-Point Protocol (PPP). (STANDARD -> STD0051)
- RFC1662 PPP in HDLC-like Framing. (STANDARD -> STD0051)

Urządzenia i ich systemy wbudowane

O ile rynek rozwiązań typowo komputerowych wydaje się być dość dojrzały i posiada dość bogata ofertę to w przypadku rozwiązań "sprzętowych" (czyli dedykowanego sprzetu wraz z działającym na nim oprogramowaniem, np. kamer IP) sytuacja wygląda znacznie gorzej. Poniżej kilka słów refleksji odnoszących się właśnie do kamer IP, ale podobne (a niekiedy większe) problemy nękają rynek sprzetowej telefoni IP, systemów automatyki budynkowej, central alarmowych, a także tańszych zarządzalnych switchy i routerów oraz wielu innych zastosowań wbudowanych.

Dobra kamera IP kosztuje kilkukrotnie (tak koło 5 - 6 razy) więcej od dobrej kamery analogowej. W przypadku tańszych rozwiązań jest to albo niezła kamera (przetwornik, tryb nocny, podświetlenie, konfigurowalne opcje DSP - takie jak balans bieli, zoom optyczny) z badziewnym IP, albo niezłe IP (streaming, uplod FTP, obsługa IPv6, ssh lub chociażby telnet do konfiguracji, ...) z badziwną kamerą (trafia się też badziewna kamera z badziewnym IP).

Zmora jest też sztuczne ograniczanie użytkownika w tym co może zrobić z jego sprzętem (nierzadko podyktowane chęcią sprzedawania zrożnicowanej funkcjonalnie gamy produktów na bazie tego samego sprzętu). Producenci bardzo często zapominają o najistotniejszej funkcjonalności (streamingu RTP / RTSP), także (często) ignorowana jest kwestia streamingu audio wraz z obrazem (ładują na siłe wifi niemalże do każdego modelu, ale na moduł mikrofonu to już szkoda funduszy / miejsca w obudowie). Z pozostałych funkcji najistotniejsza wydaje się obsługa alarmów (zarówno ich generowania w oparciu o detekcję ruchu, przyjmowania z zewnętrznego źródła, jak i wystawiania na zewnątrz bądź wykonywania akcji takiej jak upload wideo).

Dlatego też bardzo brakuje (a brak ten jest bolesny) przyzwoitego, open-sourcowy, nieograniczający użytkownika systemu na który składają się:

- podcentralki alarmowe IP - kilka(naście) wejść cyfrowych (dla klasycznych czujek ruchu, dymu), wejście protokularne 1-wire (dla termometrów), wejście protokularne RS-485 (dla czujek z komunikacją protokularną (modbus) z obsługą podpisywania odpowiedzi - odpowiedz oprócz samego statusu zawiera także zahaszowany status i sekret czujki), także wariant(y) z dodatkowym przełączanym (między 232 a 485) portem RS, portami i2c, usb, m-bus (może w formie karty rozszerzeń na i2c lub spi) ...
- opisanych protokularnych czujek (PIR, PIR + mikrofala, stłuczenie, dym optyczny, dym aktywny, zalanie, ...) i klawiatur protokularnych (jako bardziej zaawansowane mogą być stosowane urządzenia IP)
- przyzwoite kamer analogowe (noc-dzień, wymienne obiektywy, różne warianty obudowy - kopłkowa, pistoletowa, zewnętrzna, ... z IR, bez IR) i te same kamery z wbudowanym modułem IP (streaming, detekcja ruchu, wejście i wyjście alarmu, zapis na ftp/sftp, trap snmp)
- gamy wideoserwerów i wideorejestratorów IP dla kamer analogowych oraz sprzętowych kart DVR
- telefonów IP (z wsparciem dla wiadomosci tekstowych xmpp, smtp, rss, ...)

Kolejną tego typu dziedziną gdzie mamy sporo rozwiązań, niekiedy nawet niezłych, ale praktycznie zawsze z jakimiś istotnymi brakami i zawsze z sztucznie ograniczonymi możliwościami są systemy SP / RSC / RAC (Out-of-band management[2224]). Często oferują wiele kanałów dostępu (ssh, ipmi, snmp, www, ... - co jest bardzo miłe) jednak kanały te nie są równoprawne nawet w zakresie w którym bezproblemowo mogłyby być (np. odczyty wartości czujników temperatur i pracy wentylatorów, etc). Rozwiązania te często bazują na linuxach, więc wprowadzenie np. prawdziwych tagowanych VLANów (z możliwością należenia do kilku z osobną adresacją) powinno być problemowe (a chyba w żadnym z rozwiązań nie jest możliwe, ze względu na ograniczenia nakładki nałożonej na linuxową podstawę). Pewną bolączką jest też brak standaryzacji takich rozwiązań (np. CLI). Dobry SP powinien cechować się co najmniej:

- dostęp / obsługiwane protokoły:
  - CLI poprzez SSH (RFC4250 - RFC4254) oraz port szeregowy
  - IPMI 2.0
  - SNMP (Agent i Trapa)
  - WWW
- konsola zdalna:
  - konsola tekstowa pre-OS oraz OS (Remote Serial Console, ...)
  - historia konsoli tekstowej (conajmniej 60 linii)
  - konsola graficzna (Virtual KVM, VC, ...) wspierającą standardowy protokół VC

- zarządzanie zasilaniem serwera (włączenie, wyłączenie i reset zdalny)
- monitoring
  - odczyt poprzez CLI, IPMI i SNMP wartości:
    - pobieranej mocy lub prądu
    - temperatury powietrza i CPU (CPU and chassis environment)
    - prędkości wentylatorów
    - napięć +12V +5V +3.3V, baterii, ...
  - logowanie zdarzeń związanych z sensorami, stanem zasilania itp wraz z opcją wysyłania logów do zdalnego syslog serwera
  - możliwość ustawienia powiadomień SMTP (RFC 5321) i e-mail
- możliwość udostępniania dwóch wirtualnych napędów (z zdalnych napędów, obrazów ISO (Virtual CD/DVD-ROM) lub zasobu NFS) oraz bootowania z niego
- obsługa tagowanych VLANów (co najmniej dwa tagowane lub tagowany i nietagowany z niezależną adresacją IPv4 i IPV6)
- jeżeli jednostka zarządzająca posiada niezależny zegar pobieranie informacji o czasie poprzez NTP (RFC 1305) lub SNTP (RFC 4330)

Digital Video Broadcasting - czy to ma przyszłość ?

Koncepcja cyfrowej telewizji (DTV[2225]) główną popularność zyskała w drugiej połowie lat 90-tych ubiegłego wieku, niejako na tle "cyfrowej rewolucji" i rewolucji internetowej. Niestety (a może na szczęście ... ale o tym na koniec) rynek elektroniki użytkowej (w odróżnieniu od informatycznego ... przynajmniej jeszcze w tamtych latach) cechuje dość duża bezwładność i cyfrowa telewizja zaczęła docierać pod strzechy stosunkowo niedawno, natomiast zanim zastąpi liczące już grubo ponad pół wieku stosowane obecnie standardy telewizji analogowej minie przynajmniej 10 lat. Uzasadnione wydaje się zatem postawienie pytania czy w świetle rozwoju sieci pakietowych (opartych o IP), wzrostu ich przepustowości i coraz lepszym dostosowaniu do świadczenia usług multimedialnych, inwestowanie w infrastrukturę telewizji cyfrowej w standardach DVB[2226] ma sens ?

Oczywiście standardy telewizji cyfrowej (DVB-*) są wielkim krokiem na przód w stosunku co do obecnie stosowanego analogu czy nawet standardów DVD - rozdzielczość HDTV[2227] (1920x1080) może robić wrażenie, podobnie jak cyfrowa jakość przekazu, kilka wersji ścieżki dźwiękowej. Jednak jak sama nazwa skazuje nadal jest to system z ginącej już w zasadzie rodziny systemów rozgłoszeniowych (broadcasting - jeden nadaje do wszystkich), co nawet pomimo istnienia kanału zwrotnego znacznie ogranicza interakcyjność i możliwość zindywidualizowania oferty. Zresztą z powodu braku kanału zwrotnego małą popularnością cieszą się usługi DVB-MPH[2228] określane mianem telewizji interaktywnej, a tak naprawdę polegające na zunifikowanym API dla aplikacji dostarczanych poprzez kanał DVB do odbiornika telewizyjnego lub Set-top_box[2229] i na nim uruchamianych.

Również same standardy DVB-* nastręczają problemów przy wdrażaniu - odbiornik musi być dedykowany dla danego medium dostępowego (telewizji satelitarnej, kablowej, naziemnej, bądź naziemnej dla urządzeń mobilnych), gdyż dla każdego z nich istnieje odrębny standard nadawania nie zgodny z innymi (odpowiednio DVB-S, DVB-C, DVB-T, DVB-H), ma to oczywiście uzasadnienie techniczne, ale jest koszmarne od strony użytkowej (dla przykładu mając kartę w standardzie Ethernet[2230] możemy wpiąć się w gniazdko w prawie każdej sieci komputerowej). Oprócz tego można spotkać się z różnym kodowaniem źródłowym MPEG2, MPEG4 część 2, MPEG-4 AVC[2231](H.264 = JVT = arkusz 10 MPEG-4). Ponadto DVB nie jest również ogólnoświatowym standardem telewizji cyfrowej (nadal mamy sytuację podobną jak za czasów PAL[2232] i NTSC[2233] że w różnych rejonach mamy niekompatybilne ze sobą systemy telewizji) - oprócz niego są także: ISDB[2234], ATSC[2235].

Kolejnym problemem jest wielka trudność w budowaniu zbiorczych instalacji satelitarnych - w przypadku DVB-S są w praktyce dwie możliwości:

- budowa instalacji 4 przewodowej gdzie każdy odbiornik podłączany jest przez multiswitch (w odróżnieniu od instalacji ethernetowej nie ma możliwości podłączenia kolejnego switcha doi wyjścia odbiornikowego),
- budowa instalacji opartej na przemiennikach pierwszej pośredniej częstotliwości satelitarnej (IF) umożliwiający wprowadzenie do instalacji zbiorczej sygnałów wybranych przez operatora transponderów satelitarnych (w zakresie 950-2150 MHz, pasmo 46-862 MHz używane m.in. w analogu zostaje wolne).

Kolejne rozwiązania owocują nie dostarczaniem cyfrowego sygnału DVB do gniazdek telewizyjnych - jest to odbiór i transmisja tych programów jako analog (najbardziej bezsensowne rozwiązanie na DTV) bądź (chyba

najbardziej przyszłościowe rozwiązanie) odbiór i cyfrowa redystrybucja programów telewizyjnych w szybkiej sieci LAN[2236] (zanim DVB stanie się popularnym standardem powszechne będą najprawdopodobniej już sieci 10Gbit/s) w technologii multicast[2237] dla standardowych programów telewizyjnych (odbiera ten kto chce) i unicast[2238] dla usług interaktywnych (dla pojedynczego odbiorcy - w szczególności video na życzenie(VoD[2239])).

Właśnie ze względu na rozgłoszeniowy charakter DVB oraz ze względu na gigantyczny wzrost przepustowości sieci komputerowych wydaje się iż przyszłość leży właśnie w transmisjach multicastowych (sygnału odbieranego z DVB bądź nawet bezpośrednio z łącza Internetowego a nie w budowie skomplikowanych sieci DVB. Biorąc to pod uwagę nawet technologia przetworników IF/IF zbliżająca DVB-S do obecnych norm telewizyjnych (jeden przewód rozgałęziany przez stosowny osprzęt) nie wydaje sie przyszłościowa i dużo bardziej wartościowe wydaje się inwestowanie w szybką multimedialną infrastrukturę LAN, która obok pełnionych już funkcji telekomunikacyjnych (Internet - WWW, poczta, …, telefonia - VoIP) będzie też pełnić funkcję medium transmisyjnego dla wideo.

Innym problemem z DVB - głównie w wykonaniu telewizji kablowych (DVB-C) jest ograniczanie użytkownika w wyborze sprzętu służącego do odbioru (praktycznie zawsze trzeba korzystać z dekodera dostarczanego przez operatora). Nawet jeżeli posiadany odbiornik obsługuje DVB-C to i tak operator wymusza korzystanie ze swojego dekodera poprzez kodowanie programów i nie oferowanie samych kart lub kart z modułami CI oraz parowanie kart dostępu z dostarczanym dekoderem. Ponadto dostarczane przez operatorów kablowych sprzęty są mocno ograniczone pod względem funkcjonalnym (brak możliwości sterowania innego niż poprzez pilota - a jest gniazdko ethernet więc nic nie stoi na przeszkodzie w użyciu np. SNMP, brak możliwości zgrania nagrań na inny nośnik, brak możliwości odtwarzania nagrań z zewnętrznych nośników, itd.), a użytkownik do wyboru ma po jednym modelu dekodera PVR, HD i SD.

Na dodatek DVB nie jest jedynym ogólnoświatowym standardem telewizji cyfrowej - poza Europą (gdzie dominuje właśnie DVB) stosowane są też conajmniej 3 inne rodziny standardów (a każdy z nich - podobnie jak DVB występuje także w rożnych odmianach).

Zobacz w Sieci: Telewizja satelitarna[2240], Telewizja kablowa[2241], Test card[2242], DVB-Zone[2243], Cyfrowy program Dipola[2244], Instalacje wykorzystujące przemianę w zakresie pierwszej pośredniej częstotliwości satelitarnej[2245], Schemat instalacji TV naziemnej i satelitarnej w bloku z wykorzystaniem przemiany pierwszej pośredniej satelitarnej IF/IF[2246], Zbiorcze instalacje satelitarne[2247], IPTV[2248], Kajdanki na oglądanie telewizji - koszmar czy nasza przyszłość?[2249].

Zobacz także w Wikipedii: Rekomendacja CCIR 601[2250] i Rekomendacja CCIR 709[2251].

ABC inżynierii komputerowej, czyli co każdy komputerolog wiedzieć powinien

W głównym dziale tego serwisu zebrane są informacje stanowiące swego rodzaju elementarz komputerologi[2252]. Mogą być one przydatne zarówno podczas samokształcenia jak i podczas organizacji oraz realizacji zajęć w postaci bardziej zorganizowanej - np. przedmiotu wprowadzającego do kursu zaawansowanej informatyki i elektroniki dla uczniów szkół średnich (15-16 lat). Poniżej pozwolę sobie przedstawić założenia organizacyjne dla takich zajęć.

Kurs powinien mieć charakter wykładowo-warsztatowy (wykład z licznymi demonstracjami praktycznymi, które uczestnicy kursu mogą ćwiczyć bezpośrednio na laptopach …) i być zaliczany projektami realizowanymi poza czasem zajęć wykładowych (czas pracy własnej i konsultacji w laboratorium). Kurs składa się z następujących bloków po 3h (4 godziny lekcyjne, mogą być realizowane jako 2x2):

1. koncepcja systemu elektroniczno-programowego - logika 2 wartościowa i bramki logiczne, magistrala, adresy
2. KOMPUTERY
3. podstawy użytkowania UNIX - komendy, pliki, logi, urządzenia, interfejsy
4. KOMUNIKACJA - sieci TCP/IP wraz routingiem, podstawowe usługi w sieciach IP, interfejsy komunikacyjne oraz sieć ethernet
5. C - stdin/stdout, zmienne, pętle, instrukcje wyboru, wskaźniki
6. a) odrobinkę o funkcjach bibliotecznych i systemowych w C
   b) C++ - obiektowość, szablony, biblioteka standardowa
7. PODSTAWY ELEKTRONIKI - elementy bierne, dioda
8. TRANZYSTORY - bipolarne (praca jako klucz i w stanie aktywnym), mosfet

9. a) UKŁADY PROGRAMOWALNE
   b) JĘZYKI OPISU SPRZĘTU - VHDL
   c) MODELOWANIE AUTOMATÓW
10. BASH - programowanie w oparciu o aplikacje, podstawy sed i awk
11. a) AWK - przetwarzanie tekstowych baz danych
    b) INNE JĘZYKI PROGRAMOWANIA - przegląd (xml+xsl, sql, php, perl, python, ...)
12. PODSUMOWANIE - wstęp do komputerologi
13. administracja UNIX z usługami sieciowymi (w tym routingiem)
14. oprogramowanie mikroprocesora (asembler) oraz realizacja peryferiów w FPGA (VHDL), jako element inteligentnego domu
15. omówienie przykładowych zadań projektowych

Końcowe bloki (po podsumowaniu) realizowane mogą być w zależności od posiadanego czasu, z tego samego względu ich długość może być różna od nominalnej. Takie rozłożenie czasowe pozwala na realizacje tych zajęć w formie normalnego przedmiotu szkolnego jak i kursu wakacyjnego (np. dwa tygodnie, dwu godzinne zajęcia po 2 lub 3 razy dziennie)

W przypadku realizacji szkolnej zaliczenie odbywać powinno się w oparciu o dwa projekty (projekty te powinny być też realizowane w przypadku realizacji nieszkolnych):

1. (kilka prostych zagadnień programistycznych rozwiązywanych zarówno przy pomocy C/C++ jak i bash/awk
2. prosty układ elektroniczny (migacz diody) - realizowany zarówno w wersji dyskretnej (układ RC, jakaś bramka logiczna, tranzystor, ...) jak i w oparciu o mikrokontroler (dodatkowo: obsługa portu szeregowego i przerwań)

W przypadku gdy jest to przedmiot wprowadzający do większej ilości bardziej zaawansowanych zajęć (takich jak: "administracja UNIX i sieci TCP/IP", "programowanie w językach C i C++", "elektronika cyfrowa i systemy mikroprocesorowe", "(tranzystorowa) elektronika analogowa") powinien być dla nich "zalecanym poprzednikiem".

W przypadku realizacji na zasadzie samokształcenia wspomaganego współpracą z kimś bardziej obeznanym (model uczeń-mistrz) realizacja kolejnych tematów powinna odbywać się wg następującego algorytmu: 1) Wskazanie i wprowadzenie do zagadnienia. 2) Samodzielne opanowanie zagadnienia. 3) Przedstawienie wątpliwości dotyczących zagadnienia lub rozmowa egzaminacyjna. 4) W zależności od efektu pkt 3 przejście do następnego zagadnienia lub powrót do punktu 2 z podaniem dodatkowych wskazówek.

Należy jednak zawsze pamiętać iż komputerologii nie można zostać nauczonym - jej trzeba nauczyć się samemu poprzez eksperymentowanie, zgłębianie poszczególnych zagadnień itd. Mimo to przydatna jest pomoc kogoś doświadczonego, kto coś podpowie, a przede wszystkim będzie służył radą i pomocą przy trudniejszych zagadnieniach.

Bugi

Bug[2253] jest to termin określający błąd w systemie komputerowym (najczęściej oprogramowaniu). Poniżej zamieszczam listę najbardziej uciążliwych z mojego subiektywnego punktu widzenia. Jest prawdopodobne że w miarę możliwości będę się próbował przyczynić do ich eliminacji. Ponadto eliminacja poszczególnych pozycji z tej listy będzie wpływać na treści i rozwiązania prezentowane w niniejszym serwisie.

Sieci i usługi sieciowe

- wsparcie IPv6 w aplikacjach (głównie VoIP oraz klienci XMPP), wzrost jego popularności
  - fail2ban: please support ipv6 addresses (and ipv6tables)[2254]
  - psi: please enable ipv6[2255], korzystanie z IPv6 w Psi wymaga zmiany opcji konfiguracyjnej DNS i rekompilacji programu (zobacz informację[2256] na forum Psi), ale nawet pomimo to transfer plików nasłuchuje tylko na adresie IPv4 (0.0.0.0:52610).
  - VLC: brak wsparcia IPv6 w funkcji klienta RTSP[2257] (obejście poprzez `avio://rtsp://`)
  - Live555: brak wsparcia IPv6
- popularyzację rekordów SRV (obsługa w klientach rekordów dla http, smtp itp) - patrz też: zgłoszenie błędu dla Firefoxa[2258], Take a minute to improve the Internet![2259]
- lepszą implementację XMPP w aplikacjach (zarówno serwerach jak i klientach), w szególności:
  - historia wiadomości przechowywana po stronie serwera
  - komunikacja głosowa i wideo oraz przesył plików z użyciem Jingle

  - możliwość ustawienia automatycznego forwardu wiadomości między kontami oraz reazlizacja elastycznych aliasów (na wrór virtusertable)
- Option for not limiting maximum length of file names[2260] w module "fancyindex" serwera Nginx

Multimedia

- libav: problem z kopiowaniem rtsp/h264 do mpegts[2261]
- libav: problem z odczytem rtsp od vlc[2262]
- libav: "Application provided invalid, non monotonically increasing dts to muxer in stream"[2263]
- VLC: przewijanie na timeshifcie (w przód i tył)[2264]
- problemy związane z współpracą m.in. VLC z bibliotekami ffmpeg z debian-multimedia.org (np. odtwarzanie mjpeg) / wymogi aplikacji z debian-multimedia.org używania tamtejszej wersji ffmpeg (patrz też vlc-nox: ffmpeg encoder error: cannot find encoder MPEG-4 Video[2265])

Formaty danych i podstawowe narzędzia

- wymiana danych pomiędzy GEDA+PCB, KTechLab oraz KiCAD - patrz geda:format_translation[2266]
- tr doesn't support multibyte strings[2267]
- cannot start screen from a su session[2268] (tmux nie ma tego problemu)
- arping should use alternatives to remove conflict with iputils-arping[2269]
- aptitude: Display archive-name|origin|Label in package view (add %t to Package-Display-Format)[2270]
- Build the package with the OpenSSH LDAP public key patch (openssh-lpk)[2271]
- Please drop non-UTF8 locales[2272]
- libc6: strftime %s wrong result when struct tm is from gmtime_r() and timezone is non UTC[2273]

Środowiska graficzne

- Openbox:
  - option to disable event_cancel_all_key_grabs() for execute action [PATCH][2274]
  - Apply actions for new windows.[2275]
  - client list menu limited to the current desktop [PATCH][2276]
  - Support window size and position relative to screen size[2277]
- xfce4-terminal:
  - Make URL handler button configurable via hidden setting[2278]
  - Command-line colours opt.[2279]
  - Ctrl +/- to zoom[2280]
  - alternative config file[2281]
  - vte, hence Gnome-terminal, does not support blinking text[2282]
- KDE (głównie Konqueror):
  - w pewnych przypadkach (jakich?) dziwne/błędne odświeżanie w Kate 4.4.5 głównie gdy włączone jest dynamiczne łamanie linii, podobne efekty w Konqueror 4.4.5 - bugi w kde: 206746[2283]
  - brak możliwości ustawienia aby ctrl+whell w Konquerorze powiększało rozmiar czcionki a nie skalowało stonę (+ możliwość ustawienia skalowania na shift+ctrl+whell) - bugi w kde: [2284], [2285], 176069[2286], 34692[2287]
  - przeciaganie mysza tekstu w formularzach www w Konquerorze - Bug 151672 - Input lines and text areas do not accept drag and drop[2288]
  - submenu "create new" w menu kontekstowym Konqueror'a dostępne także gdy zaznaczone obiekty
  - możliwość zmiany lepkości zaznaczenia w Konquerorze (w szczególności po wklejeniu, rename, itd odznacza się)
  - problem rozwalania znaków nie ASCII gdy edytor kgpg szyfruje plik UTF-8 z bardzo dziwnymi znaczkami (sam gnupg czy nawet kgpg operujący na plikach tego nie ma)

Pozostałe

- pdfedit - add support for import and manipulation with images[2289]
- pełne wspacie 3D w R6xx patrz: x.org - RadeonFeature[2290], x.org - radeonhd[2291], x.org - radeon[2292]
- xserver-xorg-video-siliconmotion: several problems (mouse,screen) after upgrading to lenny[2293] + xserver-xorg-video-siliconmotion: [Upstream Fixes] 3 possibilities of Blank Screen and performance issue[2294]
- brak funkcjonalności wcinania złamanych części długich wciętych linii w vim-ie (patrz: VimBreakIndent[2295], Vim and Breakindent[2296])
- włączenie możliwości przełączania paneli i/lub okien w tmuxie przy pomocy myszki blokuje przewijanie kółkiem myszki w programach oraz zaznaczanie tekstu myszą i wklejanie przy użyciu kółka (patrz: Mouse issue[2297])

O systemie ochrony własności intelektualnej słów kilka

Celem systemu prawnej ochrony własności intelektualnej (zarówno prawa autorskiego, jak i patentowego) pierwotnie było i nadal powinno być zachowanie równowagi pomiędzy prawem autora do wynagrodzenia za swoją pracę a prawem ludzkości do korzystania z wspólnego dorobku kulturalnego całej cywilizacji. Pierwotny okres obowiązywania majątkowych praw autorskich wynoszący maksymalnie 28 lat od opublikowania dzieła ("Copyright Act of 1790[2298]" zapewniał wstępny okres ochrony 14 lat z możliwością przedłużenia na kolejne 14 lat) zapewniał taką równowagę. Jednak współcześnie obowiązujące 75 lat od śmierci autora zdecydowanie przedkłada interes jednostki nad dobro wspólne, utrudniając wymianę idei i naturalny ewolucyjny rozwój cywilizacji; czego przykładem może być chociażby to iż stworzenie czegokolwiek większego bez armii prawników zajmujących się zagadnieniami "własności intelektualnej" stało się praktycznie niemożliwe. Swoistym paradoksem jest także fakt iż w znacznie powolniejszym świecie wystarczający był stosunkowo krótki okres ochrony, a dzisiaj - na początku XXI pierwszego wieku gdy informacja obiega całą Kulę Ziemską w przeciągu sekund, gdzie mody i trendy zmieniają się co kilka lat okres ten osiągnął tak gigantyczną wartość. Regulacja taka prowadzi do tego iż za życia pojedynczego człowieka do domeny publicznej[2299] nie wejdą utwory, filmy, gry które pamięta z swojego dzieciństwa, mało tego brak takiego przejścia w połączeniu z coraz bardziej pędzącym światem doprowadza do tego iż wiele z nich staje się praktycznie całkowicie nie osiągalnych już po 20-30 latach.

Dlatego też uważam iż w trosce o dobro naszej cywilizacji okres ochrony majątkowych praw autorskich powinien zostać skrócony do maksymalnie 20 lat od opublikowania utworu (tak jak ma to miejsce w przypadku systemu patentowego), ponadto wprowadzony i realizowany powinien być obowiązek przekazywania kopi każdego dzieła (np. pod rygorem utraty majątkowych praw autorskich) do stosownych instytucji bibliotecznych lub archiwów, które po wygaśnięciu praw majątkowych miałyby obowiązek bezpłatnego udostępniania tych zasobów poprzez Internet.

Zobacz też: L. Lessig, Wolna kultura, Prawo: czas trwania[2300], Dzień domeny publicznej - materiały[2301], Czy piraci uratują kulturę[2302], Zagadnienie własności intelektualne na progu XXI wieku[2303].

Linki

Uwaga: bardzo wiele użytecznych odnośników znajduje się bezpośrednio w artykułach dotyczących poszczególnych zagadnień.

- The Unix and Internet Fundamentals HOWTO[2304]
- Wikipedia[2305]: Wikiportal:Informatyka[2306], Kategoria:Informatyka[2307], Kategoria:Telekomunikacja[2308], Kategoria:Elektronika[2309], Kategoria:Elektrotechnika[2310]
- Wikibooks[2311] - wolne podręczniki, m.in. do C, C++, LaTeXa, Debiana, elektroniki, ...
- Helionica[2312] - sieciowa encyklopedia informatyki (fork działu informatycznego z Wikipedii PL)
- Komputer i Sieci na Wydziale Fizyki UW[2313] (skrypt z spora dawką ciekawych i przydatnych informacji o komputerach i sieciach) oraz Pracownia komputerowa na Wydziale Fizyki UW[2314] (skrypt z podstawowymi (ale tak nie do końca ...) informacjiami o komputerach)
- Art of Assembly Language Programming and HLA by Randall Hyde[2315] - podręczniki wprowadzające do asemblera oraz zagadnień związanych z (niskopoziomowym) działaniem komputera (EN)
- Weird Science[2316] - nauki przyrodnicze w doświadczeniach

— LinuxFocus online magazine[2317]
— linux.skryptoteka.pl[2318] - zbiór podręczników manuali i dokumentacji dotyczącej Linux'a
— Archiwum "Linux Magazine"[2319]
— Dragonia Magazine[2320]

— The Linux Documentation Project[2321]
— Artykuły @ Dokumentacja Gentoo Linux[2322]
— Dokumentacja PLD Linux[2323]
— Polskie centrum BSD[2324]
— IBM's resource for developers[2325]
— Witold Paluszynski Home Page[2326]
— Dydaktyka KANiUP.eaiie.agh.edu.pl[2327]
— Nieco teorii z katalogu ELFA[2328]
— Opracowania Stowarzyszenia Energetyków Polskich[2329]
— Nie tylko fonia i CW[2330] - książka o tematyce łączności krótkofalowej
— Inżynieria programowania kart inteligentnych[2331]
— materaiły dydaktyczne MIM UW[2332] - m.in. z informatyki
— ebooks-it.org[2333]

— 7thguard[2334](portal poświecony tematyce wolnego oprogramowania w szerszym kontekście)
— Demokracja bezpośrednia, bez zdobywania większości[2335]
— Wolna Kultura[2336]
— Good Copy Bad Copy[2337]

— Tom's Hardware Guide (pl)[2338]
— PCLab.pl[2339]

— Debian[2340]
— Debian User Gang[2341]
— Linuksowo.pl - Polskie Forum Linuksowe[2342]
— Polskie Forum Linuksowe[2343]
— Polskie forum Debiana[2344]
— Free Software Fundation[2345]
— Katalog wolnego oprogramowania[2346]
— Alioth[2347] i Source Forge[2348] - hosting dla wolnego oprogramowania

— www.bsd.org[2349]
— linux.pl[2350]
— unixy.pl[2351]
— Linux w Polsce dla każdego[2352]
— Linux Storm[2353]
— OpenPoland.org[2354]
— Dział Utrzymania Ruchu - Centrum Onkologii Bydgoszcz[2355]
— Prace Pod Napięciem[2356]
— Statyka budowli[2357]
— Wytrzymałość materiałów[2358]

— Webowy klient Jabbera (i nie tylko)[2359]
— WWW shell[2360] ;-) a pozatym sporo porad dotyczących komend, pisania skryptów itd i kolejna podobna strona (macierzysta skryptu) - `http://www.masswerk.at/`
— zbiór narzędzi typu looking-glass / traceroute / ping dostępnych z www[2361]
— Gray-World.net Team[2362] - shell po HTTP
— klient jabbera w excelu (wymaga tunelu ssh ...)[2363]
— porównywarka dostawców VoIP[2364]
— SecViz - Graph Exchange[2365]
— okvm - Open source KVM over IP technology[2366]

— commons.wikimedia.org[2367] - liczne zdjęcia na wolnych licencjach (z dużym udziałem własności publicznej)
— Wikipedia:Research_resources[2368], Wikipedia:Public domain image resources[2369], ... - zbiory linków do materiałów PD lub na wolnych licencjach (wiele materiałów graficznych z tych stron znajdziemy w zasobach commons.wikimedia.org)
— Zbiór modeli LDraw zestawów LEGO[2370]

— Teleman[2371] - najlepszy w poskiej sieci program telewizyjny (udostępnia m.in. wygodny widok godzinowy i pełne archiwum)

— pomysły na reformę kalenarza, niestety chyba bez większych szans na wprowadzenie :-(- International Fixed Calendar[2372], Symmetry454[2373], World Calendar[2374]

Copyright

Przypisy

[1]http://pl.wikipedia.org/wiki/Haker (slang komputerowy)
[2]http://www.opcode.eu.org//Introduction/schemat.sch
[3]http://pl.wikipedia.org/wiki/Inteligencja (psychologia)
[4]http://pl.wikipedia.org/wiki/Algorytm
[5]http://pl.wikipedia.org/wiki/KISS (reguła)
[6]http://www.opcode.eu.org//
[7]http://pl.wikipedia.org/wiki/Komputerologia
[8]http://pl.wikipedia.org/wiki/Elektronika
[9]http://pl.wikipedia.org/wiki/Prąd elektryczny
[10]http://pl.wikipedia.org/wiki/Napięcie elektryczne
[11]http://pl.wikipedia.org/wiki/Energia elektryczna
[12]http://pl.wikipedia.org/wiki/Natężenie prądu elektrycznego
[13]http://pl.wikipedia.org/wiki/Rezystancja
[14]http://pl.wikipedia.org/wiki/Masa (elektryczność)
[15]http://pl.wikipedia.org/wiki/Impedancja
[16]http://pl.wikipedia.org/wiki/Teoria obwodów
[17]http://en.wikipedia.org/wiki/Network analysis (electrical circuits)
[18]http://teoriaobwodow.edu.pl
[19]http://pl.wikipedia.org/wiki/Opornik
[20]http://pl.wikipedia.org/wiki/Kondensator
[21]http://pl.wikipedia.org/wiki/Cewka
[22]http://zstux.ita.pwr.wroc.pl/janusz/Materialy/Podstawy tel_lab/styki_cwiczenie.pdf
[23]http://en.wikipedia.org/wiki/Arc suppression
[24]http://en.wikipedia.org/wiki/Contact protection
[25]http://en.wikipedia.org/wiki/Snubber
[26]http://pl.wikipedia.org/wiki/Dioda
[27]http://pl.wikipedia.org/wiki/Dioda prostownicza
[28]http://pl.wikipedia.org/wiki/Dioda Zenera
[29]http://pl.wikipedia.org/wiki/LED
[30]http://pl.wikipedia.org/wiki/Dioda półprzewodnikowa
[31]http://pl.wikipedia.org/wiki/Złącze p-n
[32]http://pl.wikipedia.org/wiki/Tranzystor
[33]http://pl.wikipedia.org/wiki/Tranzystor bipolarny
[34]http://pl.wikipedia.org/wiki/Tranzystor bipolarny#Stany_pracy
[35]http://koral.ise.pw.edu.pl/ burd/dz-bjt.pdf
[36]http://koral.ise.pw.edu.pl/ burd/ppracy25.pdf
[37]http://pl.wikipedia.org/wiki/Tranzystor polowy
[38]http://pl.wikipedia.org/wiki/Tranzystor polowy złączowy
[39]http://pl.wikipedia.org/wiki/IGFET
[40]http://pl.wikipedia.org/wiki/MOSFET
[41]http://pl.wikipedia.org/wiki/CMOS
[42]http://pl.wikipedia.org/wiki/IGBT
[43]http://pl.wikipedia.org/wiki/Tyrystor
[44]http://pl.wikipedia.org/wiki/Triak

[45]http://www.elenota.pl/
[46]http://www.opcode.eu.org//electronics/symbole1.sch
[47]http://www.elportal.pl/pdf/k01/32_13.pdf
[48]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc04
[49]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc08
[50]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc32
[51]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc86
[52]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc00
[53]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc20
[54]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc30
[55]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc03
[56]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc132
[57]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc02
[58]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc266
[59]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc112
[60]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc175
[61]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc273
[62]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc377
[63]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc574
[64]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc564
[65]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc125
[66]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc126
[67]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc14
[68]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc245
[69]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc139
[70]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc138
[71]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc42
[72]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc257
[73]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc153
[74]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc251
[75]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc148
[76]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc149
[77]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc165
[78]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc164
[79]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc194
[80]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc393
[81]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc4020
[82]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc4040
[83]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc193
[84]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc4066
[85]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc4051
[86]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc4052
[87]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc4053
[88]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc373
[89]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc573
[90]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc85
[91]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=hc123
[92]http://pl.wikipedia.org/wiki/Seria 7400
[93]http://en.wikipedia.org/wiki/List of 4000 series integrated circuits
[94]http://en.wikipedia.org/wiki/List of 7400 series integrated circuits
[95]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=MAX485
[96]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=MAX232
[97]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=MCP23008
[98]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=MCP23009
[99]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=PCF8574A
[100]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=PCF8591
[101]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=PCA9540
[102]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=PCA9542
[103]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=PCA9544
[104]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=PCA9548
[105]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=6N136
[106]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=6N137
[107]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=4N25
[108]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=4N35
[109]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=MOC3023
[110]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=MOC3063
[111]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=BC337
[112]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=BC327
[113]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=IRFZ44N
[114]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=BS170
[115]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=IRF9Z34N
[116]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=BF245
[117]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=BTA16
[118]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=HD44780
[119]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=DS18B20

[120]http://www.elenota.pl/search.php?man=–wszyscy–\&ile=50\&start=0\&szuk=NE555
[121]http://www.elenota.pl/search.php?man=–wszyscy–\&ile=50\&start=0\&szuk=LM358
[122]http://www.elenota.pl/search.php?man=–wszyscy–\&ile=50\&start=0\&szuk=TL081
[123]http://www.elenota.pl/search.php?man=–wszyscy–\&ile=50\&start=0\&szuk=LM239N
[124]http://en.wikipedia.org/wiki/Intel 8255
[125]http://en.wikipedia.org/wiki/Intel 8259
[126]http://pl.wikipedia.org/wiki/b:Wprowadzenie do elektroniki
[127]http://www.ibiblio.org/kuphaldt/electricCircuits/
[128]http://www.ibiblio.org/kuphaldt/socratic/sinst/
[129]http://ea.elportal.pl/
[130]http://www.elektronika24.pl
[131]http://elportal.pl/kopalnia-skarbow/
[132]http://ep.com.pl/archiwum.html
[133]http://layer.uci.agh.edu.pl/ maglay/wrona/index.php?go=dydakt
[134]http://www.slcj.uw.edu.pl/pe/
[135]http://ise.pl/forum/viewtopic.php?t=8659
[136]http://mydiy.pl/jak-znalezc-usterke-w-lampkach-choinkowych
[137]http://www.cyfronika.com.pl/kityAVT/avt139pdf.pdf
[138]http://layer.uci.agh.edu.pl/maglay/wrona/pl/podstrony/spintronix/Prezentacje/Obudowy_ukladow_scalonych.pdf
[139]http://www.elektroda.pl/
[140]http://mirley.firlej.org/artykuly
[141]http://www.kosagi.com/w/index.php?title=Novena_Main_Page
[142]http://en.wikipedia.org/wiki/b:Signals and Systems
[143]http://en.wikipedia.org/wiki/b:Microprocessor Design
[144]http://en.wikipedia.org/wiki/b:Control Systems
[145]http://en.wikipedia.org/wiki/b:Embedded Systems
[146]http://en.wikipedia.org/wiki/b:Robotics
[147]http://en.wikipedia.org/wiki/b:How To Assemble A Desktop PC
[148]http://pl.wikipedia.org/wiki/Wyświetlacz
[149]http://pl.wikipedia.org/wiki/Wyświetlacz ciekłokrystaliczny
[150]http://pl.wikipedia.org/wiki/Organiczna dioda elektroluminescencyjna
[151]http://en.wikipedia.org/wiki/Chip carrier
[152]http://en.wikipedia.org/wiki/b:Digital Electronics
[153]http://en.wikipedia.org/wiki/b:Electronics Handbook
[154]http://en.wikipedia.org/wiki/b:Analogue Electronics
[155]http://en.wikipedia.org/wiki/b:Learn Electronics
[156]http://en.wikipedia.org/wiki/b:Electronics Communication
[157]http://en.wikipedia.org/wiki/b:Electronics Fundamentals
[158]http://en.wikipedia.org/wiki/b:Fundamental Digital Electronics
[159]http://en.wikipedia.org/wiki/b:Digital Circuits
[160]http://en.wikipedia.org/wiki/b:Embedded Control Systems Design
[161]http://en.wikipedia.org/wiki/b:Systems Theory
[162]http://en.wikipedia.org/wiki/b:Analog and Digital Conversion
[163]http://en.wikipedia.org/wiki/b:Electric Motors And Generators
[164]http://en.wikipedia.org/wiki/b:Semiconductors
[165]http://en.wikipedia.org/wiki/b:Signal Processing
[166]http://en.wikipedia.org/wiki/b:Engineering Tables
[167]http://en.wikipedia.org/wiki/b:Basic Electrical Generation and Distribution
[168]http://pl.wikipedia.org/wiki/Kategoria:Elektronika
[169]http://pl.wikipedia.org/wiki/Kategoria:Elektroakustyka
[170]http://pl.wikipedia.org/wiki/Kategoria:Optoelektronika
[171]http://pl.wikipedia.org/wiki/commons:Category:Control_rooms
[172]http://pl.wikipedia.org/wiki/commons:Category:Electronics
[173]http://pl.wikipedia.org/wiki/commons:Category:Computers
[174]http://koral.ise.pw.edu.pl/ burd/hybr-p.pdf
[175]http://pl.wikipedia.org/wiki/Efekt Millera
[176]http://en.wikipedia.org/wiki/Current limiting
[177]http://pl.wikipedia.org/wiki/Elektronika cyfrowa
[178]http://pl.wikipedia.org/wiki/Dwójkowy system liczbowy
[179]http://pl.wikipedia.org/wiki/Algebra Boole'a
[180]http://pl.wikipedia.org/wiki/Dwójkowy system liczbowy
[181]http://pl.wikipedia.org/wiki/Kod 1 z n
[182]http://pl.wikipedia.org/wiki/Kod Graya
[183]http://pl.wikipedia.org/wiki/Kod Johnsona
[184]http://pl.wikipedia.org/wiki/Kod uzupełnień do dwóch
[185]http://pl.wikipedia.org/wiki/Kod znak-moduł
[186]http://pl.wikipedia.org/wiki/Kod uzupełnień do jedności
[187]http://pl.wikipedia.org/wiki/Liczba zmiennoprzecinkowa
[188]http://pl.wikipedia.org/wiki/Bramka logiczna
[189]http://pl.wikipedia.org/wiki/Alternatywa
[190]http://pl.wikipedia.org/wiki/Koniunkcja (logika)
[191]http://pl.wikipedia.org/wiki/NOT
[192]http://pl.wikipedia.org/wiki/NOR
[193]http://pl.wikipedia.org/wiki/Dysjunkcja (logika)
[194]http://pl.wikipedia.org/wiki/Alternatywa wykluczająca

[195]http://pl.wikipedia.org/wiki/XNOR
[196]http://www.randomwraith.com/logic.html
[197]http://pl.wikipedia.org/wiki/Bramka trójstanowa
[198]http://pl.wikipedia.org/wiki/Otwarty kolektor
[199]http://pl.wikipedia.org/wiki/Przerzutnik
[200]http://pl.wikipedia.org/wiki/Przerzutnik typu RS
[201]http://pl.wikipedia.org/wiki/Przerzutnik typu D
[202]http://pl.wikipedia.org/wiki/Przerzutnik typu JK-MS
[203]http://pl.wikipedia.org/wiki/Multiplekser
[204]http://pl.wikipedia.org/wiki/Demultiplekser
[205]http://pl.wikipedia.org/wiki/Pamięć cyfrowa
[206]http://pl.wikipedia.org/wiki/PLD (elektronika)
[207]http://pl.wikipedia.org/wiki/Matryca CCD
[208]http://pl.wikipedia.org/wiki/Transistor-transistor logic
[209]http://pl.wikipedia.org/wiki/CMOS
[210]http://pl.wikipedia.org/wiki/Architektura ARM
[211]http://pl.wikipedia.org/wiki/Architektura MIPS
[212]http://pl.wikipedia.org/wiki/X86
[213]http://pl.wikipedia.org/wiki/PLD (elektronika)
[214]http://pl.wikipedia.org/wiki/Hardware Description Language
[215]http://pl.wikipedia.org/wiki/VHDL
[216]http://pl.wikipedia.org/wiki/Verilog
[217]http://pl.wikipedia.org/wiki/Mikroprocesor
[218]http://pl.wikipedia.org/wiki/SPLD
[219]http://pl.wikipedia.org/wiki/PLE (elektronika)
[220]http://pl.wikipedia.org/wiki/PAL (elektronika)
[221]http://pl.wikipedia.org/wiki/GAL (elektronika)
[222]http://pl.wikipedia.org/wiki/Programmable Logic Arrays (PLA)
[223]http://pl.wikipedia.org/wiki/CPLD
[224]http://pl.wikipedia.org/wiki/Field-programmable gate array
[225]http://pl.wikipedia.org/wiki/Atmel AVR
[226]http://pl.wikipedia.org/wiki/Mikrokontroler PIC
[227]http://tuxgraphics.org/electronics/200905/embedded-tcp-ip-stack.shtml
[228]http://www.freemodbus.org/
[229]http://pl.wikipedia.org/wiki/Architektura ARM
[230]http://www.ethernut.de/en/software/
[231]http://www.freertos.org/
[232]http://www.uclinux.org/
[233]http://en.wikipedia.org/wiki/UIP (micro IP)
[234]http://www.contiki-os.org/
[235]http://www.olsonet.com/
[236]http://www.lochraster.org/etherrape/?en
[237]http://pl.wikipedia.org/wiki/Sterownik PLC
[238]http://pl.wikipedia.org/wiki/Deterministyczny automat skończony
[239]http://en.wikipedia.org/wiki/ECos
[240]http://openwrt.org/
[241]http://www.linux-mips.org/
[242]http://pl.wikipedia.org/wiki/Procesor sygnałowy
[243]http://pl.wikipedia.org/wiki/PowerPC
[244]http://pl.wikipedia.org/wiki/X86
[245]http://en.wikipedia.org/wiki/Single-board computer
[246]http://en.wikipedia.org/wiki/PC/104
[247]http://pl.wikipedia.org/wiki/System wbudowany
[248]http://pl.wikipedia.org/wiki/I²C
[249]http://pl.wikipedia.org/wiki/Universal Serial Bus
[250]http://pl.wikipedia.org/wiki/OpenWrt
[251]http://pl.wikipedia.org/wiki/Raspberry Pi
[252]http://pl.wikipedia.org/wiki/System-on-a-chip
[253]http://pl.wikipedia.org/wiki/Wolne i otwarte oprogramowanie
[254]http://en.wikipedia.org/wiki/Banana Pi
[255]http://www.banana-pi.com/eacp_view.asp?id=64
[256]http://en.wikipedia.org/wiki/BeagleBoard
[257]http://en.wikipedia.org/wiki/Cubieboard
[258]https://www.olimex.com/Products/OLinuXino/open-source-hardware
[259]http://solid-run.com/freescale-imx6-family/hummingboard/
[260]http://www.orangepi.org/
[261]http://store.propox.com/index.php?p2118
[262]http://store.propox.com/index.php?p2119
[263]http://www.ftdichip.com/Products/ICs/FT4232H.htm
[264]http://store.propox.com/index.php?p2752
[265]http://en.wikipedia.org/wiki/Libusb
[266]http://pl.wikipedia.org/wiki/Open Hardware
[267]http://pl.wikipedia.org/wiki/Open Graphics Project
[268]http://en.wikipedia.org/wiki/OpenSPARC
[269]http://pl.wikipedia.org/wiki/RepRap

[270]http://pl.wikipedia.org/wiki/OpenMoko
[271]http://pl.wikipedia.org/wiki/LEON
[272]http://en.wikipedia.org/wiki/Single-board computer
[273]http://en.wikipedia.org/wiki/Backplane
[274]http://en.wikipedia.org/wiki/Computer-on-module
[275]http://pl.wikipedia.org/wiki/ISA
[276]http://en.wikipedia.org/wiki/PC/104
[277]http://en.wikipedia.org/wiki/CompactPCI
[278]http://en.wikipedia.org/wiki/PCI eXtensions for Instrumentation
[279]http://en.wikipedia.org/wiki/VMEbus
[280]http://en.wikipedia.org/wiki/VME eXtensions for Instrumentation
[281]http://en.wikipedia.org/wiki/PICMG
[282]http://en.wikipedia.org/wiki/KIM-1
[283]http://en.wikipedia.org/wiki/ETX (form factor)
[284]http://en.wikipedia.org/wiki/COM Express
[285]http://en.wikipedia.org/wiki/XTX
[286]http://pl.wikipedia.org/wiki/Mikrokontroler
[287]http://pl.wikipedia.org/wiki/Atmel AVR
[288]http://www.propox.com/download/docs/MMnet02_pl.pdf
[289]http://www.fischl.de/usbasp/
[290]http://www.elektroda.pl/rtvforum/topic1115734.html
[291]http://www.elektroda.pl/rtvforum/topic2359783.html
[292]http://elm-chan.org/works/avrx/report_e.html
[293]http://www.ethernut.de/en/software/
[294]http://www.ethernut.de/en/documents/debian.html
[295]http://www.opcode.eu.org//digital/programowalne/nut_first.c
[296]http://pl.wikipedia.org/wiki/Architektura MIPS
[297]http://www.openwrt.org
[298]http://www.raspberrypi.org
[299]http://pl.wikipedia.org/wiki/Procesor sygnałowy
[300]http://pl.wikipedia.org/wiki/Jednostka arytmetyczno-logiczna
[301]http://pl.wikipedia.org/wiki/SIMD
[302]http://en.wikipedia.org/wiki/SPMD
[303]http://pl.wikipedia.org/wiki/Cell (procesor)
[304]http://pl.wikipedia.org/wiki/Procesor graficzny
[305]http://atmega32.republika.pl/
[306]http://radzio.dxp.pl/
[307]http://mikrokontrolery.net/
[308]http://www.pg.gda.pl/ zbczaja/
[309]http://www.linusakesson.net/scene/craft/
[310]http://www.elektroda.pl/rtvforum/viewtopic.php?t=508366
[311]http://pl.wikipedia.org/wiki/General Purpose Input/Output
[312]http://pl.wikipedia.org/wiki/Direct Memory Access
[313]http://pl.wikipedia.org/wiki/EEPROM
[314]http://pl.wikipedia.org/wiki/Universal Asynchronous Receiver and Transmitter
[315]http://pl.wikipedia.org/wiki/RS-232
[316]http://pl.wikipedia.org/wiki/EIA-485
[317]http://pl.wikipedia.org/wiki/RS422
[318]http://pl.wikipedia.org/wiki/TOSLINK
[319]http://pl.wikipedia.org/wiki/Serial Peripheral Interface
[320]http://pl.wikipedia.org/wiki/In-System Programming
[321]http://pl.wikipedia.org/wiki/AVR
[322]http://pl.wikipedia.org/wiki/I²C
[323]http://pl.wikipedia.org/wiki/1-Wire
[324]http://pl.wikipedia.org/wiki/Controller Area Network
[325]http://en.wikipedia.org/wiki/CANopen
[326]http://en.wikipedia.org/wiki/CAN Kingdom
[327]http://en.wikipedia.org/wiki/MilCAN
[328]http://en.wikipedia.org/wiki/SafetyBUS p
[329]http://pl.wikipedia.org/wiki/Universal Serial Bus
[330]http://pl.wikipedia.org/wiki/FireWire
[331]http://pl.wikipedia.org/wiki/IEEE 1284
[332]http://pl.wikipedia.org/wiki/ISA
[333]http://pl.wikipedia.org/wiki/Peripheral Component Interconnect
[334]http://pl.wikipedia.org/wiki/PCI Express
[335]http://pl.wikipedia.org/wiki/ATA (technologia)
[336]http://pl.wikipedia.org/wiki/Serial ATA
[337]http://pl.wikipedia.org/wiki/SCSI
[338]http://pl.wikipedia.org/wiki/Serial Attached SCSI
[339]http://pl.wikipedia.org/wiki/Ethernet
[340]http://pl.wikipedia.org/wiki/Model TCP/IP
[341]http://en.wikipedia.org/wiki/Fieldbus
[342]http://en.wikipedia.org/wiki/INTERBUS
[343]http://pl.wikipedia.org/wiki/Foundation Fieldbus
[344]http://pl.wikipedia.org/wiki/CC-Link

[345]http://pl.wikipedia.org/wiki/Modbus
[346]http://pl.wikipedia.org/wiki/BACnet
[347]http://pl.wikipedia.org/wiki/M-Bus
[348]http://en.wikipedia.org/wiki/Profibus
[349]http://pl.wikipedia.org/wiki/Token passing
[350]http://pl.wikipedia.org/wiki/Token ring
[351]http://en.wikipedia.org/wiki/PROFINET
[352]http://pl.wikipedia.org/wiki/S-Bus
[353]http://en.wikipedia.org/wiki/LonWorks
[354]http://pl.wikipedia.org/wiki/Inteligentny budynek
[355]http://pl.wikipedia.org/wiki/IP-852
[356]http://pl.wikipedia.org/wiki/Simple Network Management Protocol
[357]http://pl.wikipedia.org/wiki/OLE for process control
[358]http://openopc.sourceforge.net/
[359]http://pyopc.sourceforge.net/
[360]http://www.p-net.org/
[361]http://pl.wikipedia.org/wiki/Profibus
[362]http://pl.wikipedia.org/wiki/Modbus
[363]http://pl.wikipedia.org/wiki/Universal Asynchronous Receiver and Transmitter
[364]http://pl.wikipedia.org/wiki/RS-485
[365]http://pl.wikipedia.org/wiki/RS-232
[366]http://pl.wikipedia.org/wiki/M-Bus
[367]http://en.wikipedia.org/wiki/DeviceNet
[368]http://pl.wikipedia.org/wiki/CANopen
[369]http://pl.wikipedia.org/wiki/Controller Area Network
[370]http://pl.wikipedia.org/wiki/I²C
[371]http://en.wikipedia.org/wiki/Serial Peripheral Interface Bus
[372]http://pl.wikipedia.org/wiki/Ethernet
[373]http://pl.wikipedia.org/wiki/Ethernet
[374]http://bacnet.sourceforge.net/
[375]http://pes.free.fr/
[376]http://pl.wikipedia.org/wiki/Układ kombinacyjny
[377]http://pl.wikipedia.org/wiki/Układ sekwencyjny
[378]http://pl.wikipedia.org/wiki/NKB
[379]https://code.google.com/archive/p/eqntott/downloads
[380]http://pl.wikipedia.org/wiki/Automat skończony
[381]http://pl.wikipedia.org/wiki/Deterministyczny automat skończony
[382]http://pl.wikipedia.org/wiki/Automat Moore'a
[383]http://pl.wikipedia.org/wiki/Graf (matematyka)
[384]http://pl.wikipedia.org/wiki/Automat Mealy'ego
[385]http://pl.wikipedia.org/wiki/Graf (matematyka)
[386]http://pl.wikipedia.org/wiki/Komputer
[387]http://pl.wikipedia.org/wiki/Architektura procesora
[388]http://pl.wikipedia.org/wiki/Architektura von Neumanna
[389]http://pl.wikipedia.org/wiki/Architektura harwardzka
[390]http://pl.wikipedia.org/wiki/Mikrokontroler
[391]http://pl.wikipedia.org/wiki/Model programowy procesora
[392]http://pl.wikipedia.org/wiki/X86
[393]http://pl.wikipedia.org/wiki/IA-32
[394]http://pl.wikipedia.org/wiki/AMD64
[395]http://pl.wikipedia.org/wiki/Historia gier komputerowych (ósma generacja)
[396]http://pl.wikipedia.org/wiki/Architektura ARM
[397]http://pl.wikipedia.org/wiki/Architektura MIPS
[398]http://pl.wikipedia.org/wiki/Power Architecture
[399]http://pl.wikipedia.org/wiki/PowerPC
[400]http://en.wikipedia.org/wiki/POWER8
[401]http://pl.wikipedia.org/wiki/Cell (procesor)
[402]http://en.wikipedia.org/wiki/OpenPOWER Foundation
[403]http://pl.wikipedia.org/wiki/Historia gier komputerowych (siódma generacja)
[404]http://pl.wikipedia.org/wiki/SPARC
[405]http://en.wikipedia.org/wiki/OpenSPARC
[406]http://pl.wikipedia.org/wiki/LEON
[407]http://pl.wikipedia.org/wiki/Kolejność bajtów#Big_endian
[408]http://pl.wikipedia.org/wiki/Kolejność bajtów#Little_endian
[409]http://en.wikipedia.org/wiki/Floating-point unit
[410]https://wiki.debian.org/Multiarch/Tuples
[411]http://pl.wikipedia.org/wiki/Płyta główna
[412]http://pl.wikipedia.org/wiki/Plik:Motherboard diagram pl.svg
[413]http://pl.wikipedia.org/wiki/Procesor
[414]http://pl.wikipedia.org/wiki/Mostek północny
[415]http://pl.wikipedia.org/wiki/Pamięć operacyjna
[416]http://pl.wikipedia.org/wiki/RAM
[417]http://en.wikipedia.org/wiki/Memory controller
[418]http://pl.wikipedia.org/wiki/Mostek południowy
[419]http://pl.wikipedia.org/wiki/HyperTransport

[420]http://pl.wikipedia.org/wiki/Intel QuickPath Interconnect
[421]http://pl.wikipedia.org/wiki/Front side bus
[422]http://pl.wikipedia.org/wiki/Wieloprocesorowość
[423]http://pl.wikipedia.org/wiki/Direct Media Interface
[424]http://en.wikipedia.org/wiki/Flexible Display Interface
[425]http://pl.wikipedia.org/wiki/PCI Express
[426]http://pl.wikipedia.org/wiki/Serial ATA
[427]http://pl.wikipedia.org/wiki/Serial Attached SCSI
[428]http://pl.wikipedia.org/wiki/Universal Serial Bus
[429]http://pl.wikipedia.org/wiki/Hub USB
[430]http://pl.wikipedia.org/wiki/Digital Visual Interface
[431]http://pl.wikipedia.org/wiki/HDMI
[432]http://pl.wikipedia.org/wiki/DisplayPort
[433]http://en.wikipedia.org/wiki/VGA connector
[434]http://pl.wikipedia.org/wiki/Peripheral Component Interconnect
[435]http://pl.wikipedia.org/wiki/IEEE 1394
[436]http://pl.wikipedia.org/wiki/RS-232
[437]http://pl.wikipedia.org/wiki/Port szeregowy
[438]http://pl.wikipedia.org/wiki/PS/2 (port komunikacyjny)
[439]http://pl.wikipedia.org/wiki/IEEE 1284
[440]http://pl.wikipedia.org/wiki/Port równoległy
[441]http://pl.wikipedia.org/wiki/ATA (technologia)
[442]http://pl.wikipedia.org/wiki/ATAPI
[443]http://pl.wikipedia.org/wiki/Industry Standard Architecture
[444]http://pl.wikipedia.org/wiki/Extended Industry Standard Architecture
[445]http://pl.wikipedia.org/wiki/VESA Local Bus
[446]http://pl.wikipedia.org/wiki/PCI-X
[447]http://pl.wikipedia.org/wiki/PCMCIA
[448]http://pl.wikipedia.org/wiki/Accelerated Graphics Port
[449]http://pl.wikipedia.org/wiki/SCSI
[450]http://en.wikipedia.org/wiki/List of device bit rates
[451]http://pl.wikipedia.org/wiki/M.2
[452]http://pl.wikipedia.org/wiki/I²S
[453]http://pl.wikipedia.org/wiki/I²C
[454]http://en.wikipedia.org/wiki/System Management Bus
[455]http://pl.wikipedia.org/wiki/Karta SIM
[456]http://en.wikipedia.org/wiki/SATA Express
[457]http://en.wikipedia.org/wiki/ESATAp
[458]http://pl.wikipedia.org/wiki/ExpressCard
[459]http://pl.wikipedia.org/wiki/Thunderbolt (złącze komputerowe)
[460]http://pl.wikipedia.org/wiki/RAID
[461]http://pl.wikipedia.org/wiki/Karta dźwiękowa
[462]http://pl.wikipedia.org/wiki/Karta sieciowa
[463]http://pl.wikipedia.org/wiki/Karta graficzna
[464]http://pl.wikipedia.org/wiki/Procesor graficzny
[465]http://pl.wikipedia.org/wiki/Ethernet
[466]http://pl.wikipedia.org/wiki/BIOS
[467]http://pl.wikipedia.org/wiki/Unified Extensible Firmware Interface
[468]https://web.archive.org/web/20070221114622/http://www.fuw.edu.pl/ rwys/kis/kis.pdf
[469]http://www.hardwarebook.net/
[470]http://www.av-info.eu/
[471]http://pl.wikipedia.org/wiki/b:Podstawowe komponenty komputera
[472]http://en.wikipedia.org/wiki/Human interface device
[473]http://en.wikipedia.org/wiki/Input device
[474]http://pl.wikipedia.org/wiki/Mysz komputerowa
[475]http://pl.wikipedia.org/wiki/Klawiatura
[476]http://pl.wikipedia.org/wiki/Trackball
[477]http://pl.wikipedia.org/wiki/TrackPoint
[478]http://pl.wikipedia.org/wiki/Tablet graficzny
[479]http://pl.wikipedia.org/wiki/Dżojstik
[480]http://pl.wikipedia.org/wiki/Gamepad
[481]http://en.wikipedia.org/wiki/Output device
[482]http://pl.wikipedia.org/wiki/Monitor
[483]http://pl.wikipedia.org/wiki/Wyświetlacz ciekłokrystaliczny
[484]http://pl.wikipedia.org/wiki/Drukarka
[485]http://pl.wikipedia.org/wiki/Skaner
[486]http://pl.wikipedia.org/wiki/Ploter
[487]http://pl.wikipedia.org/wiki/Głośnik
[488]http://pl.wikipedia.org/wiki/Słuchawki
[489]http://pl.wikipedia.org/wiki/Mikrofon
[490]http://pl.wikipedia.org/wiki/Karta telewizyjna
[491]http://pl.wikipedia.org/wiki/Radio
[492]http://pl.wikipedia.org/wiki/Telewizja cyfrowa
[493]http://pl.wikipedia.org/wiki/Napęd optyczny
[494]http://pl.wikipedia.org/wiki/DVD

[495]http://pl.wikipedia.org/wiki/Nagrywarka
[496]http://pl.wikipedia.org/wiki/CD-R
[497]http://pl.wikipedia.org/wiki/CD-RW
[498]http://pl.wikipedia.org/wiki/DVD+R
[499]http://pl.wikipedia.org/wiki/DVD-R
[500]http://pl.wikipedia.org/wiki/DVD+RW
[501]http://pl.wikipedia.org/wiki/DVD-RW
[502]http://pl.wikipedia.org/wiki/Pamięć flash
[503]http://pl.wikipedia.org/wiki/Secure Digital
[504]http://pl.wikipedia.org/wiki/MicroSD
[505]http://pl.wikipedia.org/wiki/Modem
[506]http://pl.wikipedia.org/wiki/AMD64
[507]http://pl.wikipedia.org/wiki/Pamięć operacyjna
[508]http://pl.wikipedia.org/wiki/Blue-Ray
[509]http://pl.wikipedia.org/wiki/Interfejs (urządzenie)
[510]http://pl.wikipedia.org/wiki/Composite video
[511]http://pl.wikipedia.org/wiki/S-Video
[512]http://www.elenota.pl/search.php?man=−wszyscy−\&ile=50\&start=0\&szuk=ad725
[513]http://en.wikipedia.org/wiki/Phone connector (audio)
[514]http://en.wikipedia.org/wiki/Registered jack
[515]http://en.wikipedia.org/wiki/Modular connector
[516]http://pl.wikipedia.org/wiki/Bluetooth
[517]http://pl.wikipedia.org/wiki/Infrared Data Association
[518]http://pl.wikipedia.org/wiki/S/PDIF
[519]http://pl.wikipedia.org/wiki/TOSLINK
[520]http://pl.wikipedia.org/wiki/Systemy emisji sygnału telewizyjnego
[521]http://en.wikipedia.org/wiki/TV aerial plug
[522]http://pl.wikipedia.org/wiki/Kategoria:Sprzęt komputerowy
[523]http://pl.wikipedia.org/wiki/Kategoria:Komputery
[524]http://pl.wikipedia.org/wiki/Kategoria:Magistrale
[525]http://pl.wikipedia.org/wiki/Kategoria:Pamięci komputerowe
[526]http://pl.wikipedia.org/wiki/Kategoria:Nośniki danych
[527]http://pl.wikipedia.org/wiki/Kategoria:Dyski optyczne
[528]http://pl.wikipedia.org/wiki/Kategoria:Dyski magnetyczne
[529]http://en.wikipedia.org/wiki/Redundancy (engineering)
[530]http://pl.wikipedia.org/wiki/Hot plugging
[531]http://en.wikipedia.org/wiki/Out-of-band management
[532]http://en.wikipedia.org/wiki/19-inch rack
[533]http://pl.wikipedia.org/wiki/U (jednostka długości)
[534]http://en.wikipedia.org/wiki/Serial over LAN
[535]http://en.wikipedia.org/wiki/Intelligent Platform Management Interface
[536]http://en.wikipedia.org/wiki/KVM switch#KVM_over_IP_.28IPKVM.29
[537]http://pl.wikipedia.org/wiki/Virtual Network Computing
[538]http://en.wikipedia.org/wiki/Remote virtual media
[539]http://pl.wikipedia.org/wiki/Serwer kasetowy
[540]http://en.wikipedia.org/wiki/Passthrough
[541]http://en.wikipedia.org/wiki/Backplane
[542]http://pl.wikipedia.org/wiki/Direct Attached Storage
[543]http://pl.wikipedia.org/wiki/Hot plugging
[544]http://pl.wikipedia.org/wiki/Network Attached Storage
[545]http://pl.wikipedia.org/wiki/Storage Area Network
[546]http://pl.wikipedia.org/wiki/Fibre Channel
[547]http://pl.wikipedia.org/wiki/InfiniBand
[548]http://pl.wikipedia.org/wiki/Ethernet
[549]http://pl.wikipedia.org/wiki/ISCSI
[550]http://en.wikipedia.org/wiki/Fibre Channel over Ethernet
[551]http://pl.wikipedia.org/wiki/Network File System (protokół)
[552]http://pl.wikipedia.org/wiki/Server Message Block
[553]http://en.wikipedia.org/wiki/Water cooling#Computer_usage
[554]http://en.wikipedia.org/wiki/Computer cooling
[555]http://en.wikipedia.org/wiki/Military computers
[556]http://en.wikipedia.org/wiki/Rugged computer
[557]http://pl.wikipedia.org/wiki/System operacyjny
[558]http://pl.wikipedia.org/wiki/Dyspozytor
[559]http://pl.wikipedia.org/wiki/Pamięć wirtualna
[560]http://pl.wikipedia.org/wiki/Stronicowanie pamięci
[561]http://pl.wikipedia.org/wiki/Przerwanie
[562]http://pl.wikipedia.org/wiki/Tryb rzeczywisty
[563]http://pl.wikipedia.org/wiki/Tryb chroniony
[564]http://pl.wikipedia.org/wiki/Tablica wektorów przerwań
[565]http://pl.wikipedia.org/wiki/Proces (informatyka)
[566]http://pl.wikipedia.org/wiki/Algorytm bankiera
[567]http://pl.wikipedia.org/wiki/System operacyjny czasu rzeczywistego
[568]http://pl.wikipedia.org/wiki/System czasu rzeczywistego
[569]http://pl.wikipedia.org/wiki/Stronicowanie pamięci

[570]http://pl.wikipedia.org/wiki/Partycja wymiany
[571]http://en.wikipedia.org/wiki/Thrashing (computer science)
[572]http://pl.wikipedia.org/wiki/Segmentacja pamięci
[573]http://pl.wikipedia.org/wiki/Pamięć wirtualna
[574]http://pl.wikipedia.org/wiki/b:Pisanie OS
[575]http://en.wikipedia.org/wiki/Intel 8259
[576]http://en.wikipedia.org/wiki/Input/output base address
[577]http://pl.wikipedia.org/wiki/Bash
[578]http://pl.wikipedia.org/wiki/Powłoka Bourne'a
[579]http://pl.wikipedia.org/wiki/ASCII
[580]http://packages.debian.org/ncompress
[581]http://packages.debian.org/zip
[582]http://packages.debian.org/unrar-free
[583]http://packages.debian.org/unzip
[584]https://wiki.archlinux.org/index.php/Access_Control_Lists
[585]http://wazniak.mimuw.edu.pl/images/a/ae/Bsi_01_lab.pdf
[586]http://cyberelk.net/tim/software/patchutils/
[587]https://savannah.nongnu.org/projects/quilt/
[588]http://gnu.univ.gda.pl/ tomasz/prog/vi/refcard/virefcard-pl.pdf
[589]http://tnerual.eriogerg.free.fr/vim.html
[590]http://michael.peopleofhonoronly.com/vim/
[591]http://www.viemu.com/a_vi_vim_graphical_cheat_sheet_tutorial.html
[592]http://arctgx.ovh.org/lin/vim.html
[593]http://gnu.univ.gda.pl/ tomasz/prog/emacs/refcard/emacsrefcard-pl.pdf
[594]http://packages.debian.org/mew-bin
[595]http://packages.debian.org/nwrite
[596]http://fuse.sourceforge.net/
[597]http://pl.wikipedia.org/wiki/FUSE
[598]http://fakechroot.alioth.debian.org/
[599]http://packages.debian.org/pciutils
[600]http://packages.debian.org/pcmciautils
[601]http://packages.debian.org/pcmciautils
[602]http://packages.debian.org/usbutils
[603]http://pl.wikipedia.org/wiki/S.M.A.R.T. (informatyka)
[604]http://packages.debian.org/smartmontools
[605]http://packages.debian.org/etherwake
[606]http://forum.dug.net.pl/viewtopic.php?p=74203#74203
[607]http://www.pizzashack.org/rssh/
[608]http://www.debian.org/
[609]http://snapshot.debian.net/
[610]http://pl.wikipedia.org/wiki/Filesystem Hierarchy Standard
[611]http://www.google.pl
[612]http://pl.wikipedia.org/wiki/Sektor rozruchowy
[613]http://pl.wikipedia.org/wiki/Master Boot Record
[614]http://pl.wikipedia.org/wiki/Program rozruchowy
[615]http://en.wikipedia.org/wiki/GNU GRUB
[616]http://en.wikipedia.org/wiki/BIOS boot partition
[617]http://pl.wikipedia.org/wiki/Unified Extensible Firmware Interface
[618]http://en.wikipedia.org/wiki/GUID Partition Table
[619]http://en.wikipedia.org/wiki/EFI System partition
[620]http://pl.wikipedia.org/wiki/FAT32
[621]https://wiki.archlinux.org/index.php/GRUB
[622]https://wiki.archlinux.org/index.php/Master_Boot_Record
[623]https://wiki.archlinux.org/index.php/GUID_Partition_Table
[624]https://wiki.archlinux.org/index.php/Unified_Extensible_Firmware_Interface
[625]https://wiki.archlinux.org/index.php/Systemd
[626]https://wiki.archlinux.org/index.php/Systemd_FAQ
[627]https://wiki.archlinux.org/index.php/Systemd/Timers
[628]https://wiki.archlinux.org/index.php/Systemd/User
[629]https://medium.com/@johannes_gehrs/getting-started-with-systemd-on-debian-jessie-e024758ca63d
[630]https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services-and-units
[631]https://ask.fedoraproject.org/en/question/44938/can-systemd-replace-monit/
[632]http://blog.grzeszy.net/automontowanie-systemow-plikow-w-systemie-linux
[633]http://pl.wikipedia.org/wiki/GUID Partition Table
[634]http://en.wikipedia.org/wiki/BSD disklabel
[635]http://pl.wikipedia.org/wiki/Master Boot Record
[636]http://en.wikipedia.org/wiki/Apple Partition Map
[637]http://pl.wikipedia.org/wiki/RAID
[638]http://pl.wikipedia.org/wiki/LVM
[639]http://pl.docs.pld-linux.org/soft_raid.html
[640]http://andrzej.dopierala.name/2007-04-09_Instalacja_linuksa_na_raid1
[641]http://andrzej.dopierala.name/2007-04-11_Migracja_serwera_na_RAID1
[642]http://pl.docs.pld-linux.org/lvm2.html
[643]https://wiki.archlinux.org/index.php/LVM
[644]http://qref.sourceforge.net/

[645]https://www.debian.org/doc/manuals/debian-reference/
[646]http://qref.sourceforge.net/Debian/reference/ch-tutorial.pl.html#s-text-process
[647]http://qref.sourceforge.net/Debian/reference/ch-tutorial.pl.html#s-unixfile
[648]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s-cronjob
[649]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s8.6.23
[650]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s-setterm
[651]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s8.6.9
[652]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s-screen
[653]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s-net-test
[654]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s8.6.32
[655]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s-chroot
[656]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s8.6.35
[657]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s8.6.36
[658]http://qref.sourceforge.net/Debian/reference/ch-tune.pl.html#s-loginctrl
[659]http://qref.sourceforge.net/Debian/reference/ch-tune.pl.html#s-sudo
[660]http://qref.sourceforge.net/Debian/reference/ch-tune.pl.html#s-hdtune
[661]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s8.5.2
[662]http://en.wikipedia.org/wiki/Magic SysRq key
[663]http://qref.sourceforge.net/Debian/reference/ch-tips.pl.html#s8.6.8
[664]http://pl.wikipedia.org/wiki/Berkeley Software Distribution
[665]http://pl.wikipedia.org/wiki/OpenSolaris
[666]http://pl.wikipedia.org/wiki/FreeBSD
[667]http://pl.wikipedia.org/wiki/OpenBSD
[668]http://pl.wikipedia.org/wiki/NetBSD
[669]http://pl.wikipedia.org/wiki/Solaris (system operacyjny)
[670]http://pl.wikipedia.org/wiki/ZFS
[671]http://pl.wikipedia.org/wiki/RAID
[672]http://pl.wikipedia.org/wiki/LVM
[673]http://pl.wikipedia.org/wiki/OpenSolaris
[674]http://pl.wikipedia.org/wiki/Haiku (system operacyjny)
[675]http://pl.wikipedia.org/wiki/Darwin (system operacyjny)
[676]http://pl.wikipedia.org/wiki/OS X
[677]http://www.freedos.org/
[678]http://www.finnix.org/Balder
[679]https://www.debian.org/doc/manuals/debian-handbook/
[680]http://tldp.org/guides.html
[681]https://www.debian.org/doc/manuals/refcard/refcard.pl.pdf
[682]http://cb.vu/unixtoolbox.xhtml
[683]http://www.linuxdlafirm.pl/linux_podstawy/instalacja_urzadzen_pod_linuksem/identyfikacja_i_testowanie_sprzetu_pc_pod_linuksem.html
[684]http://en.wikipedia.org/wiki/b:A Quick Introduction to Unix
[685]http://en.wikipedia.org/wiki/b:Guide to Unix
[686]http://en.wikipedia.org/wiki/b:Guide to X11
[687]http://en.wikipedia.org/wiki/b:FOSS Network Infrastructure and Security
[688]http://www.commandlinefu.com/
[689]http://pl.wikipedia.org/wiki/Model TCP/IP
[690]http://pl.wikipedia.org/wiki/Protokół internetowy
[691]http://pl.wikipedia.org/wiki/Transmission Control Protocol
[692]http://pl.wikipedia.org/wiki/User Datagram Protocol
[693]http://pl.wikipedia.org/wiki/Model OSI
[694]http://pl.wikipedia.org/wiki/Hypertext Transfer Protocol
[695]http://pl.wikipedia.org/wiki/Simple Mail Transfer Protocol
[696]http://pl.wikipedia.org/wiki/Extensible Messaging and Presence Protocol
[697]http://pl.wikipedia.org/wiki/Czas życia pakietu
[698]http://pl.wikipedia.org/wiki/Real-time Transport Protocol
[699]http://pl.wikipedia.org/wiki/Real Time Control Protocol
[700]http://pl.wikipedia.org/wiki/Stream Control Transmission Protocol
[701]http://pl.wikipedia.org/wiki/Quality of service
[702]http://pl.wikipedia.org/wiki/Hierarchical Token Bucket
[703]http://pl.wikipedia.org/wiki/Class Based Queueing
[704]http://en.wikipedia.org/wiki/Type of service
[705]http://pl.wikipedia.org/wiki/Integrated Services
[706]http://pl.wikipedia.org/wiki/Resource Reservation Protocol
[707]http://pl.wikipedia.org/wiki/DiffServ
[708]http://en.wikipedia.org/wiki/Bandwidth Broker
[709]http://en.wikipedia.org/wiki/Common Open Policy Service
[710]http://pl.wikipedia.org/wiki/Maska podsieci
[711]http://pl.wikipedia.org/wiki/Unicast
[712]http://pl.wikipedia.org/wiki/Broadcast
[713]http://pl.wikipedia.org/wiki/Multicast
[714]http://pl.wikipedia.org/wiki/Anycast
[715]http://pl.wikipedia.org/wiki/Network Address Translation
[716]http://pl.wikipedia.org/wiki/Protokół trasowania
[717]http://pl.wikipedia.org/wiki/Border Gateway Protocol
[718]http://pl.wikipedia.org/wiki/b:Routing dynamiczny
[719]http://www.unikore.it/index.php/ingegneria-informatica-persone/docenti-del-corso/item/download/3348_02655a806b1482e89608528d203e27da

[720]http://nfsec.pl/hakin9/nat.pdf
[721]http://pl.wikipedia.org/wiki/Adres IP
[722]http://pl.wikipedia.org/wiki/Router
[723]http://pl.wikipedia.org/wiki/Internet Group Management Protocol
[724]http://en.wikipedia.org/wiki/Distance Vector Multicast Routing Protocol
[725]http://en.wikipedia.org/wiki/Multicast Open Shortest Path First
[726]http://en.wikipedia.org/wiki/Protocol Independent Multicast
[727]http://en.wikipedia.org/wiki/Protocol Independent Multicast
[728]http://pl.wikipedia.org/wiki/Open Shortest Path First
[729]http://pl.wikipedia.org/wiki/Real-time Transport Protocol
[730]http://rafa.eu.org/papers/mcasting.html
[731]http://pl.wikipedia.org/wiki/Szesnastkowy system liczbowy
[732]http://pl.wikipedia.org/wiki/Sieć lokalna
[733]http://pl.wikipedia.org/wiki/Internet Control Message Protocol
[734]http://pl.wikipedia.org/wiki/Dynamic Host Configuration Protocol
[735]http://download.weti.pl/Semestr%20I/Sieci%20komputerowe/Remigiusz_Pollum/IPv6%20-%20Overview.pdf
[736]http://klub.com.pl/dhcpv6/dibbler/master-thesis-pl/dhcpv6.pdf
[737]http://klub.com.pl/ip46nat/
[738]http://pl.wikipedia.org/wiki/Virtual Private Network
[739]http://pl.wikipedia.org/wiki/Network Address Translation
[740]http://pl.wikipedia.org/wiki/Hypertext Transfer Protocol
[741]http://pl.wikipedia.org/wiki/OpenPGP
[742]http://www.openssh.org/
[743]http://www.openssl.org/
[744]http://packages.debian.org/bridge-utils
[745]http://packages.debian.org/iputils-ping
[746]http://packages.debian.org/traceroute
[747]http://packages.debian.org/paris-traceroute
[748]http://packages.debian.org/tcptraceroute
[749]http://packages.debian.org/traceroute-nanog
[750]http://packages.debian.org/traceproto
[751]http://packages.debian.org/iputils-tracepath
[752]http://packages.debian.org/mtr
[753]http://packages.debian.org/mtr-tiny
[754]http://packages.debian.org/arping
[755]http://packages.debian.org/ndisc6
[756]http://packages.debian.org/ndisc6
[757]http://packages.debian.org/ndisc6
[758]http://packages.debian.org/ndisc6
[759]http://packages.debian.org/ndisc6
[760]http://packages.debian.org/ipv6toolkit
[761]http://packages.debian.org/ipv6toolkit
[762]http://packages.debian.org/ipv6toolkit
[763]http://netcat6.sourceforge.net/
[764]http://ftp.arl.mil/ftp/pub/ttcp/
[765]http://nms.csail.mit.edu/software/CM/apps/
[766]ftp://ftp.lcp.nrl.navy.mil/pub/nuttcp/
[767]http://pl.wikipedia.org/wiki/OpenDNS
[768]http://www.grzegorz.net/pliki/linpodradm.zip
[769]http://www.tu.kielce.pl/ sk/as/as-sieci-lokalne.pdf
[770]http://rafa.eu.org/papers/podstawy.html
[771]http://www.tldp.org/HOWTO/Multicast-HOWTO.html
[772]http://linuxreviews.org/howtos/networking/lartc/pl/index.html
[773]http://packetlife.net/library/cheat-sheets/
[774]http://www.pl.ipv6tf.org/
[775]http://dnsstuff.com/
[776]http://www.ripe.net/db/whois/whois.html
[777]http://pl.wikipedia.org/wiki/IPv6
[778]http://pl.wikipedia.org/wiki/IPv4
[779]http://pl.wikipedia.org/wiki/RJ-45
[780]http://en.wikipedia.org/wiki/IDC
[781]http://www.elektroda.pl/rtvforum/topic84380.html
[782]http://www.elektroda.pl/rtvforum/viewtopic.php?p=680288#680288
[783]http://pl.wikipedia.org/wiki/Institute of Electrical and Electronics Engineers
[784]http://pl.wikipedia.org/wiki/Address Resolution Protocol
[785]http://pl.wikipedia.org/wiki/Reverse Address Resolution Protocol
[786]http://pl.wikipedia.org/wiki/CSMA/CD
[787]http://pl.wikipedia.org/wiki/Ethernet
[788]http://pl.wikipedia.org/wiki/Bezprzewodowa sieć lokalna
[789]http://pl.wikipedia.org/wiki/Wi-Fi
[790]http://pl.wikipedia.org/wiki/Wired Equivalent Privacy
[791]http://pl.wikipedia.org/wiki/Wi-Fi Protected Access
[792]http://pl.wikipedia.org/wiki/Punkt dostępu
[793]http://en.wikipedia.org/wiki/Bridging (networking)
[794]http://ebtables.netfilter.org/examples/basic.html

[795]http://ebtables.netfilter.org/br_fw_ia/br_fw_ia.html
[796]http://pl.wikipedia.org/wiki/Plik:Netfilter-packet-flow.svg
[797]http://pl.wikipedia.org/wiki/Punkt dostępu
[798]http://pl.wikipedia.org/wiki/Wireless Distribution System
[799]http://pl.wikipedia.org/wiki/Wirtualna sieć lokalna
[800]http://www.linuxhorizon.ro/vlans.html
[801]http://pl.wikipedia.org/wiki/Trunking
[802]http://www.linuxhorizon.ro/bonding.html
[803]http://www.tu.kielce.pl/ sk/as/as-sieci-lokalne.pdf
[804]http://www.interklasa.pl/portal/index/subjectpages/informatyka/linuxadm.pdf?page=attachement&aid=25146&latch=0
[805]http://pl.wikipedia.org/wiki/Secure Shell
[806]http://pl.wikipedia.org/wiki/Secure copy
[807]http://pl.wikipedia.org/wiki/SSH File Transfer Protocol
[808]http://pl.wikipedia.org/wiki/Simple Mail Transfer Protocol
[809]http://pl.wikipedia.org/wiki/SMTP-AUTH
[810]http://pl.wikipedia.org/wiki/POP3
[811]http://pl.wikipedia.org/wiki/Internet Message Access Protocol
[812]http://pl.wikipedia.org/wiki/Network News Transfer Protocol
[813]http://pl.wikipedia.org/wiki/Usenet
[814]http://pl.wikipedia.org/wiki/Hypertext Transfer Protocol
[815]http://pl.wikipedia.org/wiki/World Wide Web
[816]http://pl.wikipedia.org/wiki/HTML
[817]http://pl.wikipedia.org/wiki/Domain Name System
[818]http://en.wikipedia.org/wiki/MX record
[819]http://en.wikipedia.org/wiki/SRV record
[820]http://pl.wikipedia.org/wiki/Internet Relay Chat
[821]http://pl.wikipedia.org/wiki/Wideo na życzenie
[822]http://pl.wikipedia.org/wiki/Voice over Internet Protocol
[823]http://pl.wikipedia.org/wiki/Real Time Streaming Protocol
[824]http://www.zoneminder.com/
[825]http://pl.wikipedia.org/wiki/Session Initiation Protocol
[826]http://pl.wikipedia.org/wiki/Real-time Transport Protocol
[827]http://pl.wikipedia.org/wiki/Hypertext Transfer Protocol
[828]http://pl.wikipedia.org/wiki/Uniform Resource Identifier
[829]http://pl.wikipedia.org/wiki/Publiczna komutowana sieć telefoniczna
[830]http://en.wikipedia.org/wiki/SIMPLE
[831]http://pl.wikipedia.org/wiki/b:Telefonia
[832]http://krzysiek.tele.pw.edu.pl/pdf/afcea2003.pdf
[833]http://pl.wikipedia.org/wiki/Extensible Messaging and Presence Protocol
[834]http://pl.wikipedia.org/wiki/Jabber
[835]http://pl.wikipedia.org/wiki/Jingle (protokół)
[836]http://pl.wikipedia.org/wiki/Voice over Internet Protocol
[837]http://pl.wikipedia.org/wiki/P2P
[838]http://pl.wikipedia.org/wiki/Telnet
[839]http://pl.wikipedia.org/wiki/File Transfer Protocol
[840]http://www.konqueror.org/
[841]http://www.mozilla.org/projects/firefox/
[842]https://addons.mozilla.org/firefox/2390/
[843]http://packages.debian.org/iceweasel
[844]http://lynx.isc.org/
[845]http://w3m.sourceforge.net/
[846]http://packages.debian.org/w3m
[847]http://packages.debian.org/w3m-img
[848]http://elinks.or.cz/
[849]http://links.twibright.com/
[850]http://www.netsurf-browser.org/
[851]http://packages.debian.org/netsurf-linuxfb
[852]http://www.httrack.com/
[853]http://packages.debian.org/httrack
[854]http://packages.debian.org/webhttrack
[855]http://www.gnu.org/software/wget/
[856]http://kget.sourceforge.net
[857]http://www.claws-mail.org/
[858]http://packages.debian.org/claws-mail
[859]http://packages.debian.org/claws-mail-feeds-reader
[860]http://packages.debian.org/claws-mail-address-keeper
[861]http://packages.debian.org/claws-mail-fancy-plugin
[862]http://packages.debian.org/claws-mail-trayicon
[863]http://packages.debian.org/claws-mail-smime-plugin
[864]http://packages.debian.org/claws-mail-pgpinline
[865]http://packages.debian.org/claws-mail-pgpmime
[866]http://packages.debian.org/claws-mail-attach-remover
[867]http://www.mozilla.org/projects/thunderbird/
[868]http://packages.debian.org/icedove
[869]http://www.sylpheed.org/

[870]http://claws.sylpheed.org/
[871]http://balsa.gnome.org/
[872]http://projects.gnome.org/evolution/
[873]http://kmail.kde.org/
[874]http://packages.debian.org/kmail
[875]http://packages.debian.org/kmailcvt
[876]http://knode.sourceforge.net/
[877]http://konversation.kde.org/
[878]http://akregator.kde.org/
[879]http://www.courier-mta.org/cone/
[880]http://www.washington.edu/alpine/
[881]http://www.mutt.org/
[882]http://www.gnu.org/software/mailutils/
[883]http://caspian.dotconf.net/menu/Software/SendEmail/
[884]http://packages.debian.org/sendemail
[885]http://www.procmail.org/
[886]http://catb.org/ esr/fetchmail/
[887]http://psi.affinix.com/
[888]http://www.pidgin.im/
[889]http://www.pidgin.im
[890]http://www.weechat.org/
[891]http://mcabber.com/
[892]http://www.gnu.org/software/freetalk/
[893]http://www.irssi.org/
[894]http://ekg2.org/
[895]http://sendxmpp.platon.sk/
[896]http://www.linphone.org/
[897]http://www.ekiga.org/
[898]http://kphone.sourceforge.net/
[899]http://www.twinklephone.com/
[900]http://ihu.sourceforge.net/
[901]http://www.openssh.com/
[902]http://www.videolan.org/vlc/
[903]http://www.rtmpd.com/
[904]http://packages.debian.org/crtmpserver
[905]http://www.apache.org/
[906]http://www.acme.com/software/mini_httpd/
[907]http://www.washington.edu/imap/
[908]http://dovecot.procontrol.fi/
[909]http://www.exim.org/
[910]http://www.sendmail.org/
[911]http://untroubled.org/nullmailer/
[912]http://esmtp.sourceforge.net
[913]http://msmtp.sourceforge.net
[914]http://www.openssh.com/
[915]http://jabberd.jabberstudio.org/
[916]http://jabberd.jabberstudio.org/2/
[917]http://www.ejabberd.im/
[918]http://www.mysql.org/
[919]http://www.postgresql.org/
[920]http://www.xinetd.org/
[921]http://www.ntp.org/
[922]http://www.proftpd.org/
[923]http://udpcast.linux.lu/
[924]http://www.flumotion.net/
[925]http://www.icecast.org/
[926]http://www.asterisk.org/
[927]http://dialogpalette.sourceforge.net/
[928]http://www.freeswitch.org/
[929]http://www.sipfoundry.org/sipXpbx/
[930]http://www.openpbx.org/
[931]http://www.phpmyadmin.net/
[932]http://phppgadmin.sourceforge.net/
[933]http://www.phpbb.com/
[934]http://www.squirrelmail.org/
[935]http://www.openwebmail.org/
[936]http://jwchat.org/
[937]http://code.google.com/p/ijab/
[938]http://www.webmin.com/
[939]http://www.webmin.com/index6.html
[940]http://www.platon.sk./projects/phpWebFileManager/
[941]http://ftp.isc.org/isc/dhcp/
[942]http://sourceforge.net/projects/dibbler/
[943]http://v6web.litech.org/radvd/dist/
[944]http://packages.debian.org/lpr

[945]http://www.cups.org/
[946]http://pl.wikipedia.org/wiki/ASCII
[947]http://pl.wikipedia.org/wiki/Hypertext Transfer Protocol
[948]http://pl.wikipedia.org/wiki/Simple Mail Transfer Protocol
[949]http://pl.wikipedia.org/wiki/Extensible Messaging and Presence Protocol
[950]http://pl.wikipedia.org/wiki/POP3
[951]http://pl.wikipedia.org/wiki/Internet Message Access Protocol
[952]http://pl.wikipedia.org/wiki/URL (informatyka)
[953]http://pl.wikipedia.org/wiki/Uniform Resource Name
[954]http://akson.sgh.waw.pl/ lp21543/index2.html
[955]http://siproxd.sourceforge.net/
[956]http://www.asterisk.org/
[957]http://www.iptel.org/ser/
[958]http://en.wikipedia.org/wiki/b:Computer network technologies and services
[959]http://pl.wikipedia.org/wiki/Język programowania
[960]http://pl.wikipedia.org/wiki/Asembler
[961]http://webster.cs.ucr.edu/
[962]http://pl.wikipedia.org/wiki/b:Asembler
[963]http://pl.wikipedia.org/wiki/Fortran
[964]http://pl.wikipedia.org/wiki/Ada (język programowania)
[965]http://pl.wikipedia.org/wiki/C Sharp
[966]http://pl.wikipedia.org/wiki/Java
[967]http://pl.wikipedia.org/wiki/JavaScript
[968]http://pl.wikipedia.org/wiki/Object Pascal
[969]http://pl.wikipedia.org/wiki/Objective-C
[970]http://pl.wikipedia.org/wiki/Visual Basic
[971]http://pl.wikipedia.org/wiki/SGML
[972]http://pl.wikipedia.org/wiki/XML
[973]http://pl.wikipedia.org/wiki/HTML
[974]http://pl.wikipedia.org/wiki/XHTML
[975]http://pl.wikipedia.org/wiki/Kaskadowe arkusze stylów
[976]http://pl.wikipedia.org/wiki/b:HTML
[977]http://pl.wikipedia.org/wiki/b:CSS
[978]http://pl.wikipedia.org/wiki/b:SVG
[979]http://www.xs4all.nl/ peterned/examples/cssmenu.html
[980]http://en.wikipedia.org/wiki/b:XForms
[981]http://pl.wikipedia.org/wiki/TeX
[982]http://pl.wikipedia.org/wiki/SQL
[983]http://pl.wikipedia.org/wiki/b:Bazy danych
[984]http://math.uni.lodz.pl/ pustelj/
[985]http://pl.wikipedia.org/wiki/C (język programowania)
[986]http://pl.wikipedia.org/wiki/C++
[987]http://pl.wikipedia.org/wiki/Linux
[988]http://pl.wikipedia.org/wiki/b:C
[989]http://pl.wikipedia.org/wiki/b:C++
[990]http://en.cppreference.com/w/
[991]http://www.cplusplus.com/
[992]http://logging.apache.org/log4cxx/index.html
[993]http://pl.wikipedia.org/wiki/Standard Template Library
[994]http://pl.wikipedia.org/wiki/POSIX
[995]http://pl.wikipedia.org/wiki/Komunikacja międzyprocesowa
[996]http://loqi01.eti.pg.gda.pl/pracownicy/macpi/info/SOLab/Linux-synchronizacjakomunikacjaprocesow.htm
[997]http://en.wikipedia.org/wiki/b:
[998]http://pl.wikipedia.org/wiki/b:C/Wskaźniki#Dynamiczna_alokacja_pami.C4.99ci
[999]http://pl.wikipedia.org/wiki/OpenMP
[1000]http://www.cmmsigma.eu/labs/parallel_comp/openmp_tutorial/index.pl.html
[1001]http://pl.wikipedia.org/wiki/C++11
[1002]http://en.wikipedia.org/wiki/b:GTK+ By Example
[1003]http://en.wikipedia.org/wiki/b:Qt
[1004]http://www.opcode.eu.org//programing/c_cpp/Qt_timer_subwindow_xml/Qt_timer_subwindow_xml.tgz
[1005]http://www.opcode.eu.org//programing/c_cpp/Qt_graphics_net_menu_toolbar/Qt_graphics_net_menu_toolbar.tgz
[1006]http://www.opcode.eu.org//programing/c_cpp/gui_beos_demo.tgz
[1007]http://wiki.videolan.org/LibVLC_SampleCode_SDL
[1008]http://www.opcode.eu.org//programing/c_cpp/obrazy_kontrolne.tar.gz
[1009]http://januszg.hg.pl/opengl/
[1010]http://www.clockworkcoders.com/oglsl/
[1011]http://www.lighthouse3d.com/opengl/glsl/
[1012]http://www.dimmension3.spine.pl/modules.php?name=Tutorials&wtd=show_cat&cat=opengl
[1013]http://www.spacesimulator.net/tutorials.html
[1014]http://4programmers.net/Z_pogranicza/Kurs_programowania_GLUT
[1015]http://forum.gamedev.pl/index.php/topic,754.0.html
[1016]http://www.opengl.org/documentation/specs/
[1017]http://pl.wikipedia.org/wiki/Shader
[1018]http://pl.wikipedia.org/wiki/OpenGL
[1019]http://pl.wikipedia.org/wiki/OpenGL Utility Toolkit

[1020]http://pl.wikipedia.org/wiki/OpenGL Shading Language
[1021]http://pl.wikipedia.org/wiki/b:OpenGL Shading Language
[1022]http://en.wikipedia.org/wiki/OpenCL
[1023]http://pl.wikipedia.org/wiki/OGRE
[1024]http://pl.wikipedia.org/wiki/Irrlicht
[1025]http://pl.wikipedia.org/wiki/Crystal Space
[1026]http://kursy.ddt.pl/?TutorialId=12
[1027]http://pl.wikipedia.org/wiki/b:OGRE
[1028]http://www.opcode.eu.org//programing/c_cpp/podstawa_silnika_3d/podstawa_silnika_3d.tgz
[1029]http://www.faqs.org/docs/kernel/
[1030]http://lwn.net/Kernel/LDD3/
[1031]http://www.opcode.eu.org//programing/c_cpp/ftdi_ds18b20.tar.gz
[1032]http://www.opcode.eu.org//programing/c_cpp/vkbd.tar.gz
[1033]http://www.fuw.edu.pl/ pjank/mn/
[1034]http://www.opcode.eu.org//system.system?id=sterowanie_magnetowidem
[1035]http://www.intercon.pl/ sektor/cbx/
[1036]http://www.icm.edu.pl/kdm/Programowanie
[1037]http://www.icm.edu.pl/kdm/Numeryka
[1038]http://pl.cpp.wikia.com/
[1039]http://david.tribble.com/text/cdiffs.htm
[1040]http://www.intra2net.com/en/developer/libftdi/
[1041]http://pes.free.fr/
[1042]http://en.wikipedia.org/wiki/b:Optimizing C++
[1043]http://en.wikipedia.org/wiki/b:C++ Programming As A Set Of Problems
[1044]http://www.linuxfocus.org/Polish/May2004/article335.shtml
[1045]http://pl.wikipedia.org/wiki/Address Resolution Protocol
[1046]http://www.tldp.org/LDP/abs/html/
[1047]http://www.republika.pl/dief/main.html
[1048]http://www.uw-team.org/index.php?id=arty/bash_kurs
[1049]http://en.wikipedia.org/wiki/b:Bourne Shell Scripting
[1050]http://pl.wikipedia.org/wiki/AWK
[1051]http://gnu.univ.gda.pl/ tomasz/prog/awk/refcard/awkrefcard-pl.pdf
[1052]http://gnu.univ.gda.pl/ tomasz/prog/awk/awk-intro/awk-man-b5.pdf
[1053]http://www.gnu.org/software/gawk/manual/
[1054]http://en.wikipedia.org/wiki/b:AWK
[1055]http://en.wikipedia.org/wiki/b:An Awk Primer
[1056]http://pl.wikipedia.org/wiki/Python
[1057]http://pl.wikipedia.org/wiki/b:Zanurkuj w Pythonie
[1058]http://www.openbookproject.net/thinkcs/
[1059]http://en.wikipedia.org/wiki/b:Non-Programmer’s Tutorial for Python 3
[1060]http://en.wikipedia.org/wiki/b:Non-Programmer’s Tutorial for Python 2.6
[1061]http://en.wikipedia.org/wiki/b:Think Python
[1062]http://en.wikipedia.org/wiki/b:A Beginner’s Python Tutorial
[1063]http://forum.dug.net.pl/viewtopic.php?pid=106462#p106462
[1064]http://pl.wikipedia.org/wiki/PHP
[1065]http://pl.php.net/manual/pl/
[1066]http://php.webinside.pl/
[1067]http://pl.wikipedia.org/wiki/b:Programowanie:PHP
[1068]http://en.wikipedia.org/wiki/b:PHP and MySQL Programming
[1069]http://pl.wikipedia.org/wiki/Model bazy danych
[1070]http://pl.wikipedia.org/wiki/Model relacyjny
[1071]http://pl.wikipedia.org/wiki/System zarządzania relacyjną bazą danych
[1072]http://pl.wikipedia.org/wiki/Rekord (informatyka)
[1073]http://pl.wikipedia.org/wiki/Pole (informatyka)
[1074]http://pl.wikipedia.org/wiki/Rachunek relacyjny
[1075]http://pl.wikipedia.org/wiki/Normalizacja bazy danych
[1076]http://pl.wikipedia.org/wiki/Postać normalna (bazy danych)
[1077]http://pl.wikipedia.org/wiki/SQL
[1078]http://pl.wikipedia.org/wiki/ODBC
[1079]http://dev.mysql.com/doc/refman/5.7/en/sql-syntax.html
[1080]http://en.wikipedia.org/wiki/b:MySQL
[1081]http://pl.wikipedia.org/wiki/XML
[1082]http://pl.wikipedia.org/wiki/Extensible Stylesheet Language
[1083]http://pl.wikipedia.org/wiki/XSL Transformations
[1084]http://pl.wikipedia.org/wiki/XPath
[1085]http://pl.wikipedia.org/wiki/XQuery
[1086]http://xmlstar.sourceforge.net/
[1087]http://stencel.mimuw.edu.pl/abwi/20011023/
[1088]http://en.wikipedia.org/wiki/b:XML - Managing Data Exchange
[1089]http://pl.wikipedia.org/wiki/VHDL
[1090]http://pl.wikipedia.org/wiki/Verilog
[1091]http://pl.wikipedia.org/wiki/Hardware Description Language
[1092]http://pl.wikipedia.org/wiki/FPGA
[1093]http://en.wikipedia.org/wiki/Register-transfer level
[1094]http://pl.wikipedia.org/wiki/AHDL

[1095]http://pl.wikipedia.org/wiki/SystemC
[1096]http://rawski.zpt.tele.pw.edu.pl/pl/node/30
[1097]http://www.ee.lsu.edu/v/refcard.pdf
[1098]http://gcc.gnu.org/
[1099]http://clang.llvm.org/
[1100]http://www.gnu.org/software/gawk/gawk.html
[1101]http://www.perl.org/
[1102]http://www.php.net/
[1103]http://www.python.org/
[1104]http://sourceforge.net/projects/strace/
[1105]http://subversion.tigris.org/
[1106]http://git-scm.com/
[1107]http://bazaar-vcs.org/
[1108]http://www.hping.org/
[1109]http://netcat6.sourceforge.net/
[1110]http://www.dest-unreach.org/socat/doc/socat-ttyovertcp.txt
[1111]ftp://ftp.lysator.liu.se/pub/unix/tcputils/
[1112]http://www.firebirdsql.org
[1113]http://www.firebirdnews.org/docs/fb2min_pl.html
[1114]http://treeline.bellz.org/
[1115]http://pl.wikipedia.org/wiki/Perl
[1116]http://pl.wikipedia.org/wiki/CPAN
[1117]http://pl.wikipedia.org/wiki/PHP
[1118]http://pl.wikipedia.org/wiki/Ruby
[1119]http://pl.wikipedia.org/wiki/Python
[1120]http://pl.wikipedia.org/wiki/b:Perl
[1121]http://en.wikipedia.org/wiki/b:Perl 6 Programming
[1122]http://en.wikipedia.org/wiki/Lua (programming language)
[1123]http://lua-users.org/wiki/LuaGenericCall
[1124]http://www.lua.org/manual/5.1/
[1125]http://en.wikipedia.org/wiki/b:Lua Programming
[1126]http://lua-users.org/wiki/SampleCode
[1127]http://pl.wikipedia.org/wiki/Ruby (język programowania)
[1128]http://pl.wikipedia.org/wiki/b:Ruby
[1129]http://pl.wikipedia.org/wiki/Tcl
[1130]http://pl.wikipedia.org/wiki/Tk
[1131]http://www.bin-co.com/tcl/tutorial/contents.php
[1132]http://pl.wikipedia.org/wiki/Erlang (język programowania)
[1133]http://pl.wikipedia.org/wiki/Programowanie funkcyjne
[1134]http://www.erlang.org/doc/
[1135]http://pl.wikipedia.org/wiki/GNU Octave
[1136]http://www.igf.fuw.edu.pl/ZOI/Matlab/
[1137]http://pl.wikipedia.org/wiki/DVI (plik TeX)
[1138]http://pl.wikipedia.org/wiki/OpenType
[1139]http://pl.wikipedia.org/wiki/OpenType
[1140]http://www.opcode.eu.org/ rrp/office-demo.pdf
[1141]ftp://ftp.gust.org.pl/TeX/info/lshort/polish/lshort2e.pdf
[1142]http://www.opcode.eu.org//system.system?id=cv.tex
[1143]http://www.opcode.eu.org//system.system?id=wizytowka.tex
[1144]http://www.ctan.org/tex-archive/help/Catalogue/entries/karnaugh.html
[1145]http://www.ctan.org/tex-archive/help/Catalogue/entries/dcpic.html
[1146]http://www.ctan.org/tex-archive/macros/latex/contrib/curve/
[1147]http://pl.wikipedia.org/wiki/MathML
[1148]http://gva.noekeon.org/blahtexml/index.html
[1149]http://www.localghost.at/latex2mathml/
[1150]ftp://sunsite.icm.edu.pl/pub/CTAN/help/Catalogue/catalogue.html
[1151]http://en.wikipedia.org/wiki/b:LaTeX
[1152]ftp://ftp.gust.org.pl/pub/GUST/doc/gentl-pl.pdf
[1153]http://www.immt.pwr.wroc.pl/ myszka/grafika/grafika.pdf
[1154]http://www.sapijaszko.net/pedeefy.pdf
[1155]http://www.tug.org/teTeX/tetex-texmfdist/doc/latex/mwcls/mwclsdoc.pdf
[1156]http://www.fuw.edu.pl/ kostecki/kurs_latexa.pdf
[1157]http://www.texample.net/tikz/examples/tag/diagrams
[1158]http://meta.wikimedia.org/wiki/Help:Formula#TeX
[1159]http://pl.wikipedia.org/wiki/Teoria automatów
[1160]http://pl.wikipedia.org/wiki/Teoria informacji
[1161]http://pl.wikipedia.org/wiki/Teoria grafów
[1162]http://pl.wikipedia.org/wiki/Język formalny
[1163]http://pl.wikipedia.org/wiki/Algorytm
[1164]http://pl.wikipedia.org/wiki/Teoria obliczeń
[1165]http://pl.wikipedia.org/wiki/Struktura danych
[1166]http://pl.wikipedia.org/wiki/Algorytmika
[1167]http://pl.wikipedia.org/wiki/Dowód poprawności algorytmu
[1168]http://pl.wikipedia.org/wiki/Złożoność obliczeniowa
[1169]http://pl.wikipedia.org/wiki/Logika Hoare'a

[1170]http://pl.wikipedia.org/wiki/Niezmiennik pętli
[1171]http://pl.wikipedia.org/wiki/Analiza składniowa
[1172]http://pl.wikipedia.org/wiki/Parser LR
[1173]http://en.wikipedia.org/wiki/LL parser
[1174]http://pl.wikipedia.org/wiki/Sieć neuronowa
[1175]http://pl.wikipedia.org/wiki/Sztuczna inteligencja
[1176]http://pl.wikipedia.org/wiki/Interpolacja (matematyka)
[1177]http://pl.wikipedia.org/wiki/Ekstrapolacja (matematyka)
[1178]http://pl.wikipedia.org/wiki/Logika rozmyta
[1179]http://en.wikipedia.org/wiki/b:Artificial Neural Networks
[1180]http://pl.wikipedia.org/wiki/Make
[1181]http://pl.wikipedia.org/wiki/Autotools
[1182]http://pl.wikipedia.org/wiki/Autoconf
[1183]http://pl.wikipedia.org/wiki/Automake
[1184]http://en.wikipedia.org/wiki/GNU Libtool
[1185]http://pl.wikipedia.org/wiki/CMake
[1186]http://en.wikipedia.org/wiki/SCons
[1187]http://www.scons.org/documentation.php
[1188]http://pl.wikipedia.org/wiki/Subversion
[1189]http://pl.wikipedia.org/wiki/Bazaar
[1190]http://pl.wikipedia.org/wiki/Git (oprogramowanie)
[1191]http://en.wikipedia.org/wiki/b:GCC Debugging
[1192]http://pl.wikipedia.org/wiki/Gra komputerowa
[1193]http://pl.wikipedia.org/wiki/OpenGL
[1194]http://pl.wikipedia.org/wiki/Silnik gry
[1195]http://pl.wikipedia.org/wiki/Silnik graficzny
[1196]http://pl.wikipedia.org/wiki/Wykrywanie kolizji
[1197]http://pl.wikipedia.org/wiki/Sztuczna inteligencja
[1198]http://pl.wikipedia.org/wiki/Simple DirectMedia Layer
[1199]http://pl.wikipedia.org/wiki/Wolne i otwarte oprogramowanie
[1200]http://en.wikipedia.org/wiki/Culling
[1201]http://pl.wikipedia.org/wiki/Shader
[1202]http://pl.wikipedia.org/wiki/Irrlicht
[1203]http://pl.wikipedia.org/wiki/OGRE
[1204]http://pl.wikipedia.org/wiki/Crystal Space
[1205]http://www.devmaster.net/engines/
[1206]ProFORMA by Qi Pan @ Cambridge University
[1207]http://www.gamedev.pl/
[1208]http://www.kde-apps.org/content/show.php/QShaderEdit?content=42586
[1209]http://www.opcode.eu.org//programing/more/grafika_3d_i_gry/qshaderedit-0.1.3.diff
[1210]http://pl.wikipedia.org/wiki/b:Programowanie komputerów
[1211]http://en.wikipedia.org/wiki/b:x86 Disassembly
[1212]http://pl.wikipedia.org/wiki/Biblioteka programistyczna
[1213]http://pl.wikipedia.org/wiki/Algorytm
[1214]http://en.wikipedia.org/wiki/b:Algorithms
[1215]http://pl.wikipedia.org/wiki/Kategoria:Algorytmy
[1216]http://pl.wikipedia.org/wiki/Kategoria:Algorytmy kompresji
[1217]http://en.wikipedia.org/wiki/b:Algorithm Implementation
[1218]http://pl.wikipedia.org/wiki/Struktura danych
[1219]http://pl.wikipedia.org/wiki/Kategoria:Struktury danych
[1220]http://pl.wikipedia.org/wiki/Abstrakcyjny typ danych
[1221]http://pl.wikipedia.org/wiki/B-Drzewo
[1222]http://pl.wikipedia.org/wiki/Drzewo (informatyka)
[1223]http://pl.wikipedia.org/wiki/Drzewo AVL
[1224]http://pl.wikipedia.org/wiki/Drzewo czerwono-czarne
[1225]http://pl.wikipedia.org/wiki/Binarne drzewo poszukiwań
[1226]http://pl.wikipedia.org/wiki/Kolejka (informatyka)
[1227]http://pl.wikipedia.org/wiki/Kolejka priorytetowa
[1228]http://pl.wikipedia.org/wiki/Kopiec (informatyka)
[1229]http://pl.wikipedia.org/wiki/Lista
[1230]http://pl.wikipedia.org/wiki/Rekord (informatyka)
[1231]http://pl.wikipedia.org/wiki/Stos (informatyka)
[1232]http://pl.wikipedia.org/wiki/Tablica (informatyka)
[1233]http://pl.wikipedia.org/wiki/Tablica asocjacyjna
[1234]http://pl.wikipedia.org/wiki/Tablica mieszająca
[1235]http://pl.wikipedia.org/wiki/Sortowanie
[1236]http://pl.wikipedia.org/wiki/Sortowanie bąbelkowe
[1237]http://pl.wikipedia.org/wiki/Sortowanie przez wstawianie
[1238]http://pl.wikipedia.org/wiki/Sortowanie przez scalanie
[1239]http://pl.wikipedia.org/wiki/Sortowanie przez zliczanie
[1240]http://pl.wikipedia.org/wiki/Sortowanie kubełkowe
[1241]http://pl.wikipedia.org/wiki/Sortowanie pozycyjne
[1242]http://pl.wikipedia.org/wiki/Sortowanie przez wybieranie
[1243]http://pl.wikipedia.org/wiki/Sortowanie Shella
[1244]http://pl.wikipedia.org/wiki/Sortowanie grzebieniowe

[1245]http://pl.wikipedia.org/wiki/Sortowanie szybkie
[1246]http://pl.wikipedia.org/wiki/Sortowanie introspektywne
[1247]http://pl.wikipedia.org/wiki/Sortowanie przez kopcowanie
[1248]http://pl.wikipedia.org/wiki/Metoda numeryczna
[1249]http://en.wikipedia.org/wiki/b:JPEG - Idea and Practice
[1250]http://en.wikipedia.org/wiki/b:Data Coding Theory
[1251]http://en.wikipedia.org/wiki/b:Data Compression
[1252]http://en.wikipedia.org/wiki/b:Digital Signal Processing
[1253]http://en.wikipedia.org/wiki/b:Floating Point
[1254]http://en.wikipedia.org/wiki/b:The Computer Revolution
[1255]http://en.wikipedia.org/wiki/b:Communication Networks
[1256]http://en.wikipedia.org/wiki/b:Computer Systems Engineering
[1257]http://en.wikipedia.org/wiki/b:Operating System Design
[1258]http://www.fuw.edu.pl/ pablo/s/
[1259]http://www.4programmers.net/
[1260]http://www.stelmach.info/
[1261]http://www.koders.com/
[1262]ext
[1263]http://linuxgazette.net/123/smith.html
[1264]http://pl.wikipedia.org/wiki/Prąd przemienny
[1265]http://pl.wikipedia.org/wiki/Prąd stały
[1266]http://pl.wikipedia.org/wiki/RJ-45
[1267]http://en.wikipedia.org/wiki/F connector
[1268]http://pl.wikipedia.org/wiki/BNC
[1269]http://en.wikipedia.org/wiki/RCA connector
[1270]http://pl.wikipedia.org/wiki/Topologia sieci komputerowej
[1271]http://pl.wikipedia.org/wiki/Topologia magistrali
[1272]http://pl.wikipedia.org/wiki/Topologia pierścienia
[1273]http://pl.wikipedia.org/wiki/Topologia gwiazdy
[1274]http://pl.wikipedia.org/wiki/Topologia rozgałęzionej gwiazdy
[1275]http://pl.wikipedia.org/wiki/Topologia hierarchiczna
[1276]http://en.wikipedia.org/wiki/Dropped ceiling
[1277]http://en.wikipedia.org/wiki/Raised floor
[1278]http://www.opcode.eu.org//inne/photo/trasy_kablowe.0.
[1279]http://download.czechtrade.cz/odsi.asp?id=15819
[1280]http://www.baks.com.pl/index.php?event=showpage&id=3&lang=pl
[1281]http://www.lukasiak.com.pl/pl/podlogi/
[1282]http://www.mera-system.com.pl/msl100.html
[1283]http://pl.wikipedia.org/wiki/Panel krosowniczy
[1284]http://pl.wikipedia.org/wiki/RJ-45
[1285]http://en.wikipedia.org/wiki/F connector
[1286]http://pl.wikipedia.org/wiki/BNC
[1287]http://pl.wikipedia.org/wiki/Rozdzielnica elektryczna
[1288]http://www.opcode.eu.org//installations/pukt_dystrybucyjny_10U.dxf
[1289]http://pl.wikipedia.org/wiki/Rack
[1290]http://en.wikipedia.org/wiki/19-inch rack
[1291]http://pl.wikipedia.org/wiki/U (jednostka długości)
[1292]http://pl.wikipedia.org/wiki/Szyna DIN
[1293]http://www.opcode.eu.org//inne/photo/trasy_kablowe.0.
[1294]http://pl.wikipedia.org/wiki/Szyna DIN
[1295]http://pokoj.com.pl/upload/kat/zlaczki200803.pdf
[1296]http://www.opcode.eu.org//inne/photo/elektryka_w_rack.0.
[1297]http://pl.wikipedia.org/wiki/BMS
[1298]http://pl.wikipedia.org/wiki/Wyłącznik instalacyjny
[1299]http://pl.wikipedia.org/wiki/Bezpiecznik topikowy
[1300]http://pl.wikipedia.org/wiki/Wyłącznik różnicowoprądowy
[1301]http://pl.wikipedia.org/wiki/Stycznik
[1302]http://pl.wikipedia.org/wiki/Przekaźnik
[1303]http://www.protekol.com/artykuly/20_22_Ei_0108.pdf
[1304]http://www.opcode.eu.org//installations/electrical/RG_2path_UPS_n+1.dxf
[1305]http://en.wikipedia.org/wiki/IEC 60309
[1306]http://pl.wikipedia.org/wiki/Układ sieciowy
[1307]http://pl.wikipedia.org/wiki/Transil
[1308]http://pl.wikipedia.org/wiki/Warystor
[1309]http://pl.wikipedia.org/wiki/Ogranicznik przepięć
[1310]http://pl.wikipedia.org/wiki/Listwa przeciwprzepięciowa
[1311]http://pl.wikipedia.org/wiki/Instalacja odgromowa
[1312]http://pl.wikipedia.org/wiki/Klatka Faradaya
[1313]http://pl.wikipedia.org/wiki/Układ sieciowy
[1314]http://pl.wikipedia.org/wiki/Instalacja elektryczna
[1315]http://pl.wikipedia.org/wiki/Rozdzielnica elektryczna
[1316]http://pl.wikipedia.org/wiki/Bezpiecznik elektryczny
[1317]http://pl.wikipedia.org/wiki/Bezpiecznik różnicowo-prądowy
[1318]http://pl.wikipedia.org/wiki/Gniazdo elektryczne
[1319]http://pl.wikipedia.org/wiki/Łącznik elektryczny

[1320]http://en.wikipedia.org/wiki/Electrical room
[1321]http://en.wikipedia.org/wiki/Electric switchboard
[1322]http://en.wikipedia.org/wiki/Utility tunnel
[1323]http://en.wikipedia.org/wiki/American wire gauge
[1324]http://pl.wikipedia.org/wiki/commons:Category:Industrial_lamps
[1325]http://www.ee.pw.edu.pl/sep/nowa/index.php?plik=Publikacje.html
[1326]http://www.elektrycywiejscy.irsep.org/downloads/instalacje-sklad.pdf
[1327]http://www.opcode.eu.org//inne/photo/inst_elektryczne.0.
[1328]http://pl.wikipedia.org/wiki/Inteligentny budynek
[1329]http://pl.wikipedia.org/wiki/KNX
[1330]http://pl.wikipedia.org/wiki/BACnet
[1331]http://pl.wikipedia.org/wiki/RS-232
[1332]http://pl.wikipedia.org/wiki/HTTPS
[1333]http://pl.wikipedia.org/wiki/Secure Shell
[1334]http://plc-home.pl/
[1335]http://www.zegaruz.republika.pl/
[1336]http://www.dipol.com.pl/targi/domnet.htm
[1337]http://www.eib.pl/
[1338]http://www.elmark.com.pl/index.php?id=41
[1339]http://www.elmark.com.pl/index.php?id=329
[1340]http://en.wikipedia.org/wiki/110 block
[1341]http://en.wikipedia.org/wiki/Keystone module
[1342]http://pl.wikipedia.org/wiki/System alarmowy
[1343]http://pl.wikipedia.org/wiki/Telewizja przemysłowa
[1344]http://pl.wikipedia.org/wiki/Real Time Streaming Protocol
[1345]http://pl.wikipedia.org/wiki/Hypertext Transfer Protocol
[1346]http://pl.wikipedia.org/wiki/Novec 1230
[1347]http://pl.wikipedia.org/wiki/Heptafluoropropan
[1348]http://www.opcode.eu.org//inne/photo/sys_alarmowe.0.
[1349]http://www.opcode.eu.org//inne/photo/cctv.0.
[1350]http://www.dom.pl/elektronika-drzwi-i-okna.html
[1351]http://pl.wikipedia.org/wiki/commons:Category:Fire_alarms
[1352]http://pl.wikipedia.org/wiki/commons:Category:Security_cameras
[1353]http://www.wagnerpoland.pl/fileadmin/user_upload/Bilder/Impulse/Polnisch/2006-3-pl.pdf
[1354]http://www.tycoeurope.com/fileadmin/user_upload/Poland/PDF/Exting/Inergen_folder.pdf
[1355]http://web.archive.org/web/*/http://www.sgsp.edu.pl/uczelnia/kbb/tsz/materialy/zalaczniki/gaz1.pdf
[1356]http://web.archive.org/web/*/http://www.sgsp.edu.pl/uczelnia/kbb/tsz/materialy/zalaczniki/gaz2.pdf
[1357]http://web.archive.org/web/*/http://www.sgsp.edu.pl/uczelnia/kbb/tsz/materialy/zalaczniki/gaz3.pdf
[1358]http://web.archive.org/web/*/http://www.sgsp.edu.pl/uczelnia/kbb/tsz/materialy/zalaczniki/co2.pdf
[1359]http://web.archive.org/web/*/http://www.sgsp.edu.pl/uczelnia/kbb/tsz/materialy/zalaczniki/klim.pdf
[1360]http://pl.wikipedia.org/wiki/Żelbet
[1361]http://pl.wikipedia.org/wiki/Klasa reakcji na ogień
[1362]http://pl.wikipedia.org/wiki/Szkło piankowe
[1363]http://pl.wikipedia.org/wiki/Wełna mineralna
[1364]http://pl.wikipedia.org/wiki/Wełna szklana
[1365]http://pl.wikipedia.org/wiki/Klinkier
[1366]http://pl.wikipedia.org/wiki/Granit
[1367]http://pl.wikipedia.org/wiki/Skały magmowe
[1368]http://pl.wikipedia.org/wiki/Obciążenia wyjątkowe
[1369]http://pl.wikipedia.org/wiki/Odporność ogniowa
[1370]http://www.opcode.eu.org//installations/projects_and_realization/wytyczne-serwerownia/serwerownia_koncepcja.dxf
[1371]http://www.opcode.eu.org//installations/projects_and_realization/wytyczne-serwerownia/koncepcja_szafy_dystrybucyjnej_i_adminow-ni.dxf
[1372]http://directory.fsf.org/
[1373]http://pl.wikipedia.org/wiki/Kategoria:Wolne i Otwarte Oprogramowanie
[1374]http://www.linuxsoft.cz/pl/
[1375]http://www.kde-apps.org/
[1376]http://www.kernel.org/
[1377]http://www.freebsd.org/
[1378]http://www.netbsd.org/
[1379]http://www.openbsd.org/
[1380]http://www.freedos.org
[1381]http://www.finnix.org/Balder
[1382]http://www.haiku-os.org/
[1383]http://syllable.sourceforge.net/
[1384]http://www.atheos.cx/
[1385]http://idyllaos.org/
[1386]http://www.winehq.com/
[1387]http://www.dosbox.com/
[1388]http://bochs.sourceforge.net/
[1389]http://fabrice.bellard.free.fr/qemu/
[1390]http://www.x.org/
[1391]http://packages.debian.org/xorg
[1392]http://packages.debian.org/xserver-xorg
[1393]http://www.xfree86.org/

[1394]http://www.gnome.org/
[1395]http://www.kde.org/
[1396]http://www.kde.org/
[1397]http://www.gnome.org/
[1398]http://www.kde.org/
[1399]http://clipit.rspwn.com/
[1400]http://lxde.org
[1401]http://fbpanel.sourceforge.net/
[1402]http://code.google.com/p/tint2/
[1403]http://pim.kde.org/components/knotes.php
[1404]http://openbox.org/
[1405]http://fluxbox.org/
[1406]http://www.nongnu.org/ratpoison/
[1407]http://enlightenment.org/
[1408]http://www.windowmaker.org/
[1409]http://www.xfce.org/
[1410]http://tripie.sweb.cz/utils/wmctrl/
[1411]http://sourceforge.net/projects/libxosd
[1412]http://packages.debian.org/xosd-bin
[1413]http://www.jwz.org/xscreensaver/
[1414]http://www.konqueror.org/
[1415]http://tuxcmd.sourceforge.net/
[1416]http://krusader.sourceforge.net/
[1417]http://www.krename.net/
[1418]http://utils.kde.org/projects/ark
[1419]http://kfilereplace.sourceforge.net/
[1420]http://www.kde.org/
[1421]http://p7zip.sourceforge.net/
[1422]http://packages.debian.org/p7zip-full
[1423]http://konsole.kde.org/
[1424]http://www.afterstep.org/aterm.php
[1425]http://tilda.sourceforge.net/
[1426]http://gnu.univ.gda.pl/ tomasz/prog/vi/refcard/virefcard-pl.pdf
[1427]http://tnerual.eriogerg.free.fr/vim.html
[1428]http://michael.peopleofhonoronly.com/vim/
[1429]http://www.viemu.com/a_vi_vim_graphical_cheat_sheet_tutorial.html
[1430]http://arctgx.ovh.org/lin/vim.html
[1431]http://gnu.univ.gda.pl/ tomasz/prog/emacs/refcard/emacsrefcard-pl.pdf
[1432]http://www.xemacs.org/
[1433]http://dickey.his.com/vile/vile.html
[1434]http://kate-editor.org/
[1435]http://www.tug.org/texlive/
[1436]http://www.scribus.net/
[1437]http://pdfedit.petricek.net/
[1438]http://www.ecademix.com/JohannesHofmann/flpsed.html
[1439]http://xournal.sourceforge.net/
[1440]https://www.pdflabs.com/tools/pdftk-the-pdf-toolkit/
[1441]http://www.ghostscript.com/
[1442]http://www.tardis.ed.ac.uk/ ajcd/psutils/
[1443]ftp://ftp.ics.ele.tue.nl/pub/users/jos/poster/
[1444]http://fontforge.sourceforge.net/
[1445]http://fontmatrix.net/
[1446]http://www.foolabs.com/xpdf/
[1447]http://kpdf.kde.org/
[1448]http://www.gnome.org/projects/evince/
[1449]ftp://ftp.gnu.org/gnu/gv/
[1450]http://sourceforge.net/projects/dvipng/
[1451]http://www.andre-simon.de/
[1452]http://sourceforge.net/projects/html2ps
[1453]http://dl.bytesex.org/releases/fbida/
[1454]http://freshmeat.net/projects/svp
[1455]http://www.openoffice.org/
[1456]http://www.libreoffice.org/
[1457]http://www.calligra.org/
[1458]http://www.abisource.com/
[1459]http://www.gnome.org/projects/gnumeric/
[1460]http://www.gimp.org/
[1461]http://kolourpaint.sourceforge.net/
[1462]http://www.koffice.org/krita/
[1463]http://www.cinepaint.org/
[1464]http://bourbon.usc.edu:8001/tgif/
[1465]http://mars.netanya.ac.il/ samy/tkpaint.html
[1466]http://www.blender.org/
[1467]http://www.yafray.org/
[1468]http://k3d.sourceforge.net/

[1469]http://www.kpovmodeler.org/
[1470]http://http.cs.berkeley.edu/ schenney/sced/sced.html
[1471]http://wolfpack.twu.net/Vertex
[1472]http://moonlight3d.net/
[1473]http://ayam.sourceforge.net/
[1474]http://www.inkscape.org/
[1475]http://www.koffice.org/karbon/
[1476]http://www.xfig.org/
[1477]http://www.graphviz.org/
[1478]http://www.pythoncad.org/
[1479]http://www.qcad.org/
[1480]http://pl.wikipedia.org/wiki/QCad
[1481]http://www.gnome.org/projects/dia/
[1482]http://ipe.compgeom.org/
[1483]http://www.imagemagick.org/
[1484]http://netpbm.alioth.debian.org/
[1485]http://pl.wikipedia.org/wiki/Portable anymap
[1486]http://www.greyc.ensicaen.fr/ dtschump/greycstoration/demonstration.html
[1487]http://lisas.de/kover/
[1488]http://xmorph.sourceforge.net/
[1489]http://autotrace.sourceforge.net/
[1490]http://potrace.sourceforge.net/
[1491]http://www.pstoedit.net/pstoedit
[1492]http://linuxbrit.co.uk/gozer/
[1493]http://pmt.sourceforge.net/pngcrush/
[1494]http://digikam.sourceforge.net/
[1495]http://www.flexer.it/galrey
[1496]http://altmark.nat.uni-magdeburg.de/ jschulen/ocr/
[1497]http://www.gphoto.org/proj/gtkam/
[1498]http://www.complang.tuwien.ac.at/ schani/metapixel/
[1499]http://www.gnu.org/software/panorama
[1500]http://xaos.theory.org/
[1501]http://www.xsane.org/
[1502]http://www.kde.org/
[1503]http://www.kde.org/
[1504]http://www.gphoto.org/
[1505]http://en.wikipedia.org/wiki/USB mass storage device class
[1506]http://www.kde.org/
[1507]http://heroinewarrior.com/cinelerra.php3
[1508]http://kino.schirmacher.de/
[1509]http://kdenlive.org/
[1510]http://www.openmovieeditor.org/
[1511]hhttp://www.openshotvideo.com/
[1512]http://avidemux.sourceforge.net/
[1513]http://pitivi.sourceforge.net/
[1514]http://www.xs4all.nl/ salsaman/lives/
[1515]http://www.cinefx.org/
[1516]http://www.zoneminder.com/
[1517]http://bytesex.org/xawtv/
[1518]http://kino.schirmacher.de/
[1519]http://motion.sourceforge.net/
[1520]http://www.turbolinux.com.cn/ merlin/mvc/
[1521]http://www.transcoding.org/cgi-bin/transcode
[1522]http://www.mplayerhq.hu/
[1523]http://gv4l.sourceforge.net/
[1524]http://dvbcut.sourceforge.net/
[1525]http://theorur.tvlivre.org/
[1526]http://www.bunkus.org/videotools/ogmtools/
[1527]http://www.bunkus.org/videotools/mkvtoolnix/
[1528]http://www.kraus.tk/projects/konverter/
[1529]http://mjpeg.sourceforge.net/
[1530]http://www.gnu.org/software/vcdimager/
[1531]http://www.vorbis.com/
[1532]http://audacity.sourceforge.net/
[1533]http://www.wakkanet.fi/ kaiv/ecawave/
[1534]http://ardour.org/
[1535]http://awacs.dhs.org/software/gnusound/
[1536]http://www.etse.urv.es/ dmanye/quelcom/quelcom.html
[1537]http://rezound.sourceforge.net/
[1538]http://ccrma.stanford.edu/software/snd/
[1539]http://www.metadecks.org/software/sweep/index.html
[1540]http://www.speech.kth.se/wavesurfer/
[1541]http://glame.sourceforge.net/
[1542]http://www.mtek.chalmers.se/ hjormagn/mhwaveedit.html
[1543]http://sonik.sourceforge.net/

[1544]http://kwave.sourceforge.net/
[1545]http://www.rosegardenmusic.com/
[1546]http://wavemixer.sourceforge.net/
[1547]http://www.alsa-project.org/
[1548]http://linux.bytesex.org/misc/krecord.html
[1549]http://sunsite.unc.edu/pub/Linux/apps/sound/recorder
[1550]http://kstreamripper.tuxipuxi.de/
[1551]ftp://sunsite.unc.edu/pub/Linux/apps/sound/
[1552]http://www.logarithmic.net/pfh/synaesthesia
[1553]http://mpgtx.sourceforge.net/
[1554]http://easytag.sourceforge.net/
[1555]http://prokyon3.sf.net/
[1556]http://www1.cs.columbia.edu/ cvaill/normalize/
[1557]http://timidity.sourceforge.net
[1558]http://sox.sourceforge.net
[1559]http://lame.sourceforge.net/
[1560]http://darkice.sourceforge.net/
[1561]http://www.kde.org/
[1562]http://developer.kde.org/ larrosa/kmid.html
[1563]http://www.xiph.org/paranoia/
[1564]http://www.videolan.org/vlc/
[1565]http://www.mplayerhq.hu/
[1566]http://mpv.io/
[1567]http://www.mplayer2.org/
[1568]http://kmplayer.kde.org/
[1569]http://github.com/huceke/omxplayer
[1570]http://www.xine-project.org/
[1571]http://packages.debian.org/xine-ui
[1572]http://www.xmms.org/
[1573]http://www.tf.hut.fi/ flu/cplay/
[1574]http://www.wakkanet.fi/ kaiv/ecasound/
[1575]http://www.musicpd.org/
[1576]http://rucus.ru.ac.za/ urban/projects/playmp3list/
[1577]http://splay.sourceforge.net/
[1578]http://amarok.kde.org/
[1579]http://www.gnome.org/projects/rhythmbox/
[1580]http://banshee-project.org/
[1581]http://noatun.kde.org/
[1582]http://www.listen-project.org/
[1583]http://www.beep-media-player.org/site/BMPx_Homepage
[1584]http://gwenview.sourceforge.net/
[1585]http://www.kipi-plugins.org
[1586]http://www.exiv2.org
[1587]http://imgseek.sourceforge.net/
[1588]http://www.jalix.org/projects/showimg
[1589]http://dl.bytesex.org/releases/fbida/
[1590]http://www.kde.org/
[1591]http://svg.kde.org/
[1592]http://djvu.sourceforge.net/
[1593]http://developer.kde.org/ kdvi/
[1594]http://www.kde.org/
[1595]http://kuickshow.sourceforge.net/
[1596]http://www.dtek.chalmers.se/groups/dvd/
[1597]http://freevo.sourceforge.net/
[1598]http://gphoton.sourceforge.net/
[1599]http://www.sect.mce.hw.ac.uk/ peteri/sx/
[1600]http://smplayer.sourceforge.net/en/index.php
[1601]http://tvtime.sourceforge.net/
[1602]http://linux.bytesex.org/xawtv/
[1603]http://www.goron.de/ froese/
[1604]http://www.mythtv.org/
[1605]http://xawdecode.sourceforge.net/
[1606]http://xawdecode.sourceforge.net/
[1607]http://fftv.sourceforge.net/
[1608]http://foobazco.org/projects/gradio/
[1609]http://gqmpeg.sourceforge.net/radio.html
[1610]http://www.exploits.org/v4l/fmtools/
[1611]http://utils.kde.org/projects/kgpg/
[1612]http://www.kde.org/
[1613]http://www.caffeinated.me.uk/kompare/
[1614]http://www.kde.org/
[1615]http://utils.kde.org/projects/ktimer/
[1616]http://shfs.sourceforge.net/
[1617]http://fuse.sourceforge.net/sshfs.html
[1618]http://synergy-foss.org/

[1619]http://antony.lesuisse.org/qweb/trac/wiki/AjaxTerm
[1620]http://anyterm.org/
[1621]http://monkey.org/ marius/trickle/
[1622]http://aspell.net/
[1623]http://c64.org/ nr/xwhois/
[1624]http://pim.kde.org/components/kaddressbook.php
[1625]http://utils.kde.org/projects/kcalc/
[1626]http://www.gnuplot.info/
[1627]http://www.fsf.org/software/plotutils/plotutils.html
[1628]http://labplot.sourceforge.net/
[1629]http://yacas.sourceforge.net/yacas.html
[1630]http://maxima.sourceforge.net/
[1631]http://www.octave.org/
[1632]http://www.scilab.org/
[1633]http://stardict.sourceforge.net/
[1634]http://recordmydesktop.sourceforge.net/
[1635]http://griffith.berlios.de/
[1636]http://www.partimage.org/
[1637]http://clonezilla.sourceforge.net/
[1638]http://www.kde-apps.org/content/show.php/Kvkbd?content=56019
[1639]http://www.kde-apps.org/content/show.php/?content=47371
[1640]http://www.leocad.org/linux.htm
[1641]http://klatexformula.sourceforge.net/
[1642]http://liquidweather.net/other_themes/date_calc.php
[1643]http://linux4you.h18.ru/calenculator/calenculator.en.html
[1644]http://k3b.sourceforge.net/
[1645]http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html
[1646]http://cdrdao.sourceforge.net/
[1647]https://alioth.debian.org/projects/debburn/
[1648]http://sourceforge.net/projects/linux-udf/
[1649]http://www.rastersoft.com/programas/devede.html
[1650]http://dvdisaster.net/en/
[1651]http://aptoncd.sourceforge.net
[1652]http://fy.chalmers.se/ appro/linux/DVD+RW/
[1653]http://sourceforge.net/projects/linux-udf/
[1654]http://pl.wikipedia.org/wiki/UDF (system plików)
[1655]http://spamassassin.apache.org/full/3.0.x/dist/doc/Mail_SpamAssassin_Conf.html
[1656]http://spamassassin.apache.org/tests.html
[1657]http://www.opcode.eu.org//usage_and_config/network_services/email/virtusertable
[1658]http://www.opcode.eu.org//usage_and_config/network_services/email/whitelist-hosts
[1659]exim4-spy.conf
[1660]http://en.wikipedia.org/wiki/Sieve (mail filtering language)
[1661]http://www.squirrelmail.org/
[1662]http://www.squirrelmail.org/plugins.php
[1663]http://www.squirrelmail.org/plugin_view.php?id=55
[1664]http://www.squirrelmail.org/plugin_view.php?id=73
[1665]http://www.squirrelmail.org/plugin_view.php?id=256
[1666]http://www.squirrelmail.org/plugin_view.php?id=54
[1667]http://www.squirrelmail.org/plugin_view.php?id=186
[1668]http://www.squirrelmail.org/plugin_view.php?id=120
[1669]http://www.squirrelmail.org/plugin_view.php?id=239
[1670]http://www.squirrelmail.org/plugin_view.php?id=79
[1671]http://www.squirrelmail.org/plugin_view.php?id=174
[1672]http://www.squirrelmail.org/plugin_view.php?id=112
[1673]http://www.squirrelmail.org/plugin_view.php?id=93
[1674]http://www.squirrelmail.org/plugin_view.php?id=8
[1675]http://www.squirrelmail.org/plugin_view.php?id=32
[1676]http://www.squirrelmail.org/plugin_view.php?id=195
[1677]http://www.squirrelmail.org/plugin_view.php?id=91
[1678]http://www.courier-mta.org/cone/
[1679]http://www.opcode.eu.org//usage_and_config/network_services/email/konwert_to_8bit
[1680]http://www.opcode.eu.org//usage_and_config/network_services/email/spam_check
[1681]http://www.opcode.eu.org//usage_and_config/network_services/email/spam_mv
[1682]http://www.opcode.eu.org//usage_and_config/network_services/email/notify_to_xmpp_chat
[1683]http://pl.wikipedia.org/wiki/Lista dyskusyjna
[1684]http://pl.wikipedia.org/wiki/Extensible Messaging and Presence Protocol
[1685]http://ejabberd.im/
[1686]http://www.ndl.kiev.ua/content/mod_archive_odbc-release
[1687]http://ejabberd.im/mod_archive2
[1688]http://www.opcode.eu.org//archiwum/materialy_publikowane_gdzieindziej/Patches/mod_archive_webview.diff
[1689]http://ejabberd.im/mod_multicast
[1690]https://support.process-one.net/browse/EJAB-449
[1691]http://ejabberd.im/ejabberd_ircd
[1692]http://ejabberd.im/ejabberd_users
[1693]http://ejabberd.im/mod_logxml

[1694]http://ejabberd.im/mod_filter
[1695]http://ejabberd.im/mod_asterisk
[1696]http://ejabberd.im/mod_client_asterisk
[1697]http://www.ejabberd.im/contributions
[1698]http://wiki.jrudevels.org/index.php/Eng:J2J:AdminGuide
[1699]http://www.planeta.toliman.pl/search/label/Jorge
[1700]http://www.opcode.eu.org//usage_and_config/network_services/xmpp/ejabberd.cfg
[1701]http://bepointbe.be/jabwebhist/
[1702]http://www.ejabberd.im/protocols
[1703]http://xmpp.org/extensions/
[1704]http://www.opcode.eu.org//usage_and_config/network_services/xmpp/gpg_sign_error.patch
[1705]http://forum.psi-im.org/thread/4375
[1706]http://www.opcode.eu.org//usage_and_config/network_services/xmpp/xmpp_smtp/xmpp-smtp.tgz
[1707]http://pl.wikipedia.org/wiki/Voice over Internet Protocol
[1708]http://pl.wikipedia.org/wiki/Asterisk
[1709]http://pl.wikipedia.org/wiki/Session Initiation Protocol
[1710]http://pl.wikipedia.org/wiki/Network Address Translation
[1711]http://pl.wikipedia.org/wiki/ENUM
[1712]http://pl.wikipedia.org/wiki/Extensible Messaging and Presence Protocol
[1713]http://pl.wikipedia.org/wiki/Jingle (protokół)
[1714]http://pl.wikipedia.org/wiki/G.711
[1715]http://pl.wikipedia.org/wiki/G.726
[1716]http://pl.wikipedia.org/wiki/Speex
[1717]http://pl.wikipedia.org/wiki/Kodek GSM
[1718]http://pl.wikipedia.org/wiki/Internet Low Bit Rate Codec
[1719]http://en.wikipedia.org/wiki/G722
[1720]http://www.voip-info.org/wiki/index.php?page=Asterisk
[1721]http://httpd.apache.org/docs/2.0/mod/
[1722]http://www.opcode.eu.org//usage_and_config/network_services/www/mini-httpd.conf
[1723]http://www.opcode.eu.org//usage_and_config/network_services/www/proxy-php.nginx
[1724]http://www.opcode.eu.org//usage_and_config/network_services/www/proxy.nginx
[1725]http://www.opcode.eu.org//usage_and_config/network_services/www/proxy_params.nginx
[1726]http://en.wikipedia.org/wiki/Server Name Indication
[1727]http://pl.wikipedia.org/wiki/Unicode
[1728]http://pl.wikipedia.org/wiki/Punycode
[1729]http://pl.wikipedia.org/wiki/Uniform Resource Locator
[1730]http://pl.wikipedia.org/wiki/Uniform Resource Identifier
[1731]http://pl.wikipedia.org/wiki/Internationalized Resource Identifier
[1732]http://pl.wikipedia.org/wiki/Unicode
[1733]http://pl.wikipedia.org/wiki/UTF-32/UCS-4
[1734]http://pl.wikipedia.org/wiki/UCS-2
[1735]http://pl.wikipedia.org/wiki/UTF-16
[1736]http://pl.wikipedia.org/wiki/UTF-8
[1737]http://pl.wikipedia.org/wiki/UTF-7
[1738]http://pl.wikipedia.org/wiki/Punycode
[1739]http://pl.wikipedia.org/wiki/Bom (ujednoznacznienie)
[1740]http://pl.wikipedia.org/wiki/Quoted-Printable
[1741]http://en.wikipedia.org/wiki/Percent-encoding
[1742]http://pl.wikipedia.org/wiki/Base64
[1743]http://en.wikipedia.org/wiki/Base32
[1744]http://en.wikipedia.org/wiki/Comparison of Unicode encodings
[1745]http://pl.wikipedia.org/wiki/Podpis cyfrowy
[1746]http://www.gnupg.org/
[1747]http://www.7thguard.net/news.php?id=4510
[1748]http://en.wikipedia.org/wiki/S/MIME
[1749]http://en.wikipedia.org/wiki/Server Name Indication
[1750]http://forum.dug.net.pl/viewtopic.php?pid=112994#p112994
[1751]http://groups.google.com/group/pl.comp.os.linux.sieci/browse_thread/thread/1188ed70f12a6546/fbd47e7fc3482ee4
[1752]http://meriste.net/code/patches/alter_nat/
[1753]http://pl.wikipedia.org/wiki/Point-to-Point Protocol over Ethernet
[1754]http://pl.wikipedia.org/wiki/Serwer pośredniczący
[1755]http://www.quagga.net/
[1756]http://pl.wikipedia.org/wiki/Quagga
[1757]http://pl.wikipedia.org/wiki/GNU Zebra
[1758]http://www.xorp.org/
[1759]http://bird.network.cz/
[1760]http://www.openbgpd.org/
[1761]http://fivebits.net/proj/mrd6/
[1762]http://sourceforge.net/projects/mrt/
[1763]http://freshmeat.net/projects/mrouted/
[1764]http://netweb.usc.edu/pim/pimd/
[1765]http://lwn.net/Articles/324989/
[1766]http://xkr47.outerspace.dyndns.org/netfilter/packet_flow/packet_flow10.png
[1767]http://linuxreviews.org/howtos/networking/lartc/pl/c1330.html
[1768]http://www.jukie.net/ bart/multicast/Linux-Mrouted-MiniHOWTO.html

[1769]http://en.wikipedia.org/wiki/IPv6 transition mechanism
[1770]http://www.lucastomicki.net/naptd.php
[1771]http://www.opcode.eu.org//usage_and_config/network_services/router/naptd.diff
[1772]http://pl.wikipedia.org/wiki/Martwy klawisz
[1773]http://pl.wikipedia.org/wiki/Klawisz komponujący
[1774]http://pl.wikipedia.org/wiki/Alt-kody
[1775]http://dug.net.pl/tekst/151/
[1776]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/xmodmaprc
[1777]http://www.linuxtv.org/wiki/index.php/RealTek_RTL2832U#Remote_Control
[1778]http://ftp.suse.com/pub/people/sbrabec/keyboards/input_linux_map_fixed_old.txt
[1779]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/i18n-locales-pl_PL
[1780]http://pl.wikipedia.org/wiki/Network file system
[1781]http://pl.wikipedia.org/wiki/Storage Area Network
[1782]http://pl.wikipedia.org/wiki/Fibre Channel
[1783]http://pl.wikipedia.org/wiki/Fibre Channel
[1784]http://pl.wikipedia.org/wiki/ISCSI
[1785]http://pl.wikipedia.org/wiki/Lustre (system plików)
[1786]http://pl.wikipedia.org/wiki/RAID
[1787]http://pl.wikipedia.org/wiki/Network Attached Storage
[1788]http://pl.wikipedia.org/wiki/Direct Attached Storage
[1789]http://www.freenas.org/
[1790]http://en.wikipedia.org/wiki/Fork bomb
[1791]http://en.wikipedia.org/wiki/Cgroups
[1792]http://www.brendangregg.com/linuxperf.html
[1793]http://pl.wikipedia.org/wiki/Xen
[1794]http://virt.kernelnewbies.org/TechComparison
[1795]http://wiki.debian.org/SystemVirtualization
[1796]http://pl.wikipedia.org/wiki/QEMU
[1797]http://pl.wikipedia.org/wiki/Kernel-based Virtual Machine
[1798]http://pl.wikipedia.org/wiki/VirtualBox
[1799]http://en.wikipedia.org/wiki/Linux-VServer
[1800]http://en.wikipedia.org/wiki/OpenVZ
[1801]http://www.debian-administration.org/articles/566
[1802]http://wiki.debian.org/Xen
[1803]http://www.virtuatopia.com/index.php/Xen_Virtualization_Essentials
[1804]http://www.linux.com/feature/125977
[1805]http://pl.wikipedia.org/wiki/System plików
[1806]http://pl.wikipedia.org/wiki/Ext2
[1807]http://pl.wikipedia.org/wiki/Ext3
[1808]http://pl.wikipedia.org/wiki/XFS
[1809]http://pl.wikipedia.org/wiki/ReiserFS
[1810]http://pl.wikipedia.org/wiki/Lustre (system plików)
[1811]http://www.uw-team.org/index.php?id=arty/minidist
[1812]http://www.informatik.uni-koeln.de/fai/
[1813]http://sourceforge.net/projects/linuxcreate/
[1814]http://wiki.debian.org/HowToRebuildAnOfficialDebianKernelPackage
[1815]http://pl.wikipedia.org/wiki/Ppi
[1816]http://pl.wikipedia.org/wiki/Dpi
[1817]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/xorg.conf
[1818]http://pl.wikipedia.org/wiki/VNC
[1819]http://pl.wikipedia.org/wiki/NX
[1820]http://synergy-foss.org/
[1821]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/window_show_action.sh
[1822]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/kwin.eventsrc
[1823]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/kwinrulesrc
[1824]http://ar.linux.it/software/gunzets/
[1825]http://packages.debian.org/lenny/joystick
[1826]http://pl.wikipedia.org/wiki/Universal Disk Format
[1827]http://pl.wikipedia.org/wiki/XML
[1828]http://pl.wikipedia.org/wiki/Multipurpose Internet Mail Extensions
[1829]http://standards.freedesktop.org/desktop-entry-spec/latest/
[1830]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/pdflatex.desktop
[1831]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/archiwa.tgz
[1832]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/menu.desktop
[1833]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/applications-kmenuedit.menu
[1834]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/desktop/mymenu.tgz
[1835]http://www.kde-apps.org/content/show.php/kio_rar?content=17527
[1836]http://www.kde-apps.org/content/show.php/kio_p7zip?content=17829
[1837]http://www.kde-apps.org/content/show.php/P7zip+Service+Menu?content=34414
[1838]http://www.kde-apps.org/content/show.php/Service+menus+for+Debian?content=62062
[1839]http://www.kde-apps.org/content/show.php/MountISO?content=11577
[1840]http://www.kde-apps.org/content/show.php/Open+as+user?content=36898
[1841]http://www.kde-apps.org/content/show.php/KFileManagerUtils?content=48302
[1842]http://www.kde-apps.org/content/show.php/Get+YouTube+Video+(improved)?content=41456
[1843]http://www.kde-apps.org/content/show.php/kfile_raw_patch?content=58753

[1844]http://docs.kde.org/stable/pl/kdebase/userguide/
[1845]http://www.debian.org/
[1846]http://debian.linux.org.pl/
[1847]http://www.7thguard.net/news.php?id=3231
[1848]http://www.7thguard.net/news.php?id=3112
[1849]http://www.debian.org/doc/manuals/apt-howto/
[1850]http://pl.wikipedia.org/wiki/b:Debian - uniwersalna instalacja
[1851]http://packages.debian.org/debmirror
[1852]http://forum.dug.net.pl/viewtopic.php?pid=95485
[1853]http://forum.dug.net.pl/viewtopic.php?pid=117903
[1854]http://www.feyrer.de/g4u/
[1855]http://linux.hanski.info/static/scribus/index.html
[1856]http://www.debian-administration.org/articles/420
[1857]http://linuxgazette.net/124/smith.html
[1858]http://pl.wikipedia.org/wiki/b:Linux
[1859]http://www.piotr.wolny.pl/lnag/
[1860]http://en.wikipedia.org/wiki/UnionFS
[1861]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/workenv/workenv.tgz
[1862]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/workenv/workenv.notar
[1863]http://www.opcode.eu.org//usage_and_config/operating_systems_config/linux/workenv/workenv-install.sh/workenv-install.sh
[1864]http://www.opcode.eu.org//archiwum/ikony
[1865]http://www.opcode.eu.org//archiwum/materialy_publikowane_gdzieindziej/Patches/fbpanel
[1866]http://pl.wikipedia.org/wiki/Network Information Service
[1867]http://pl.wikipedia.org/wiki/Lightweight Directory Access Protocol
[1868]http://wiki.debian.org/LDAP/NSS
[1869]http://wiki.debian.org/LDAP/PAM
[1870]http://openssh-lpk.googlecode.com/
[1871]http://www.mantisbt.org/bugs/view.php?id=11470
[1872]http://www.mantisbt.org/bugs/view.php?id=11488
[1873]http://www.mantisbt.org/bugs/view.php?id=8471
[1874]http://www.opcode.eu.org//usage_and_config/operating_systems_config/servers/passwd.php
[1875]http://en.wikipedia.org/wiki/Intelligent Platform Management Interface
[1876]http://buttersideup.com/docs/howto/IPMI_on_Debian.html
[1877]http://www.gnu.org/software/freeipmi/
[1878]http://www.opcode.eu.org//archiwum/icm/puppet/
[1879]http://www.opcode.eu.org//usage_and_config/operating_systems_config/servers/jail.conf
[1880]http://www.sshguard.net/
[1881]http://www.linux-ha.org/Heartbeat
[1882]http://www.openais.org/
[1883]http://ctdb.samba.org/
[1884]http://www.drbd.org/
[1885]http://en.wikipedia.org/wiki/Heartbeat
[1886]http://en.wikipedia.org/wiki/IPFilter
[1887]http://en.wikipedia.org/wiki/Ipfirewall
[1888]http://en.wikipedia.org/wiki/PF (firewall)
[1889]http://www.bsdua.org/netfwd.html
[1890]http://www.freebsd.org/ports/
[1891]http://freebsd.therek.net/handbook/install-steps.html#SYSINSTALL-LABEL
[1892]http://pl.wikipedia.org/wiki/Berkeley Fast File System
[1893]http://pl.wikipedia.org/wiki/Unix File System
[1894]http://dug.net.pl/tekst/144/instalacja_debiana_gnu_kfreebsd_6_0_na_zfs/
[1895]http://freebsd.therek.net/handbook/
[1896]http://www.openbsd.org/faq/
[1897]http://freebsd.therek.net/articles/new-users/article.html
[1898]http://freebsd.therek.net/articles/filtering-bridge/article.html
[1899]http://en.wikipedia.org/wiki/Write Anywhere File Layout
[1900]http://www.sun.drydog.com/faq/s86faq.html#s9.37
[1901]http://pl.wikipedia.org/wiki/Blender (program)
[1902]http://pl.wikipedia.org/wiki/OGRE
[1903]http://www.opcode.eu.org//usage_and_config/multimedia/blender/startup.blend
[1904]http://www.opcode.eu.org//usage_and_config/multimedia/blender/window1.svg
[1905]http://www.blendswap.com/blends/view/55901
[1906]http://www.blender.pl/component/option,com_wrapper/Itemid,56/index.php?option=com_content&task=view&id=396&Itemid=9
[1907]http://www.opcode.eu.org//usage_and_config/multimedia/blender/window2.svg
[1908]http://code.google.com/p/blender2ogre/
[1909]https://bitbucket.org/cc8192/blender2ogre/src
[1910]https://code.google.com/p/torchlight-to-blender/
[1911]https://code.google.com/p/torchlight-to-blender/
[1912]http://wiki.blender.org/index.php/Doc%3APL/Manual/Your_First_Animation/1.A_static_Gingerbread_Man
[1913]http://www.blender.pl/index.php?option=com_content&task=view&id=153&Itemid=9
[1914]http://www.blender.pl/index.php?option=com_content&task=view&id=38&Itemid=9
[1915]http://www.blender.pl/index.php?option=com_content&task=view&id=25&Itemid=9
[1916]http://www.blender.pl/index.php?option=com_content&task=view&id=61&Itemid=9
[1917]http://en.wikipedia.org/wiki/b:Blender 3D: Blending Into Python
[1918]http://en.wikipedia.org/wiki/b:Blender 3D: Noob to Pro

[1919]http://forum.dug.net.pl/viewtopic.php?pid=117664#p117664
[1920]http://www.linux-sound.org/dsp.html
[1921]http://www.opcode.eu.org//usage_and_config/multimedia/xawtv
[1922]http://www.goron.de/ froese/
[1923]http://www.opcode.eu.org//usage_and_config/multimedia/asound.conf
[1924]http://pl.wikipedia.org/wiki/PulseAudio
[1925]http://pl.wikipedia.org/wiki/JACK Audio Connection Kit
[1926]http://pl.wikipedia.org/wiki/MIDI
[1927]http://sourceforge.net/projects/timidity
[1928]http://forum.dug.net.pl/viewtopic.php?id=17368
[1929]http://forum.dug.net.pl/viewtopic.php?id=12622
[1930]http://jackaudio.org/applications
[1931]http://ubuntuforums.org/showthread.php?t=789578
[1932]http://pl.wikipedia.org/wiki/VLC media player
[1933]http://www.videolan.org/doc/vlc-user-guide/en/ch07.html
[1934]http://www.opcode.eu.org//usage_and_config/multimedia/streaming.sh
[1935]http://www.opcode.eu.org//usage_and_config/multimedia/mosaic.sh
[1936]http://www.rtmpd.com/
[1937]http://www.opena.tv/
[1938]http://www.opcode.eu.org//usage_and_config/multimedia/enigma2_dvb_stb/patches.tar.gz
[1939]http://liftoff.github.io/GateOne/Developer/terminal.html
[1940]http://www.opcode.eu.org//usage_and_config/multimedia/enigma2_dvb_stb/pluginTelnetClient.tgz
[1941]http://www.opcode.eu.org//usage_and_config/multimedia/enigma2_dvb_stb/FBinit.c
[1942]http://www.satsupreme.com/showthread.php/194074-Lamedb-format-explained
[1943]http://pl.wikipedia.org/wiki/SCADA
[1944]http://www.szarp.org/
[1945]http://oscada.org/
[1946]http://www.zabbix.com/
[1947]http://www.nagios.org/
[1948]http://www.cacti.net/
[1949]http://en.wikipedia.org/wiki/Comparison of network monitoring systems
[1950]http://www.zabbix.com/
[1951]http://pl.wikipedia.org/wiki/Nagios
[1952]http://en.wikipedia.org/wiki/Munin (software)
[1953]http://en.wikipedia.org/wiki/Comparison of network monitoring systems
[1954]http://en.wikipedia.org/wiki/Monit
[1955]http://www.a2s.pl/komputery-embedded-platforma-c-834_1933_107.html
[1956]http://www.opcode.eu.org//usage_and_config/monitoring/smtp2sms/install_and_run.sh
[1957]http://www.opcode.eu.org//usage_and_config/monitoring/smtp2sms/Makefile
[1958]http://www.opcode.eu.org//usage_and_config/monitoring/smtp2sms/nullmailer.build.sh
[1959]http://untroubled.org/nullmailer/
[1960]http://www.opcode.eu.org//usage_and_config/monitoring/smtp2sms/smtp2sms.tgz
[1961]http://www.a2s.pl/komputery-embedded-platforma-c-834_1933_107.html
[1962]http://www.sqlite.org/
[1963]http://www.opcode.eu.org//usage_and_config/monitoring/modbus/crc.c
[1964]http://www.opcode.eu.org//usage_and_config/monitoring/modbus/crc.h
[1965]http://www.netrino.com/Embedded-Systems/How-To/CRC-Calculation-C-Code
[1966]http://www.netrino.com/code/crc.zip
[1967]http://www.opcode.eu.org//usage_and_config/monitoring/modbus/Makefile
[1968]http://www.opcode.eu.org//usage_and_config/monitoring/modbus/install_and_run.sh
[1969]http://www.opcode.eu.org//usage_and_config/monitoring/modbus/modbus_rtu.tgz
[1970]http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf
[1971]http://www.a2s.pl/komputery-embedded-platforma-c-834_1933_107.html
[1972]http://www.opcode.eu.org//usage_and_config/monitoring/one_wire/one_wire.tgz
[1973]http://libav.org/avserver.html
[1974]http://pl.wikipedia.org/wiki/Simple Network Management Protocol
[1975]http://pl.wikipedia.org/wiki/BACnet
[1976]http://net-snmp.sourceforge.net/
[1977]http://bacnet.sourceforge.net/
[1978]http://www.opcode.eu.org//usage_and_config/monitoring/snmp-bacnet_gateway/snmp-bacnet_gateway.tgz
[1979]http://www.opcode.eu.org//archiwum/icm/snmp-bacnet_gateway_bacnet-stack-0.5.5.tgz
[1980]http://pl.wikipedia.org/wiki/EIA-485
[1981]http://pl.wikipedia.org/wiki/Modbus
[1982]http://pl.wikipedia.org/wiki/I²C
[1983]http://pl.wikipedia.org/wiki/Architektura ARM
[1984]http://www.raspberrypi.org/
[1985]http://store.propox.com/index.php?p2752,mmusb4232h
[1986]http://www.freemodbus.org/
[1987]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/czujki_alarmowe.sch
[1988]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.sch
[1989]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.pcb
[1990]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.sch
[1991]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/detekcja.pcb
[1992]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/ft4232-rs485.sch
[1993]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/ft4232-rs485.pcb

[1994]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/i2c-1wire.sch
[1995]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/i2c-1wire.pcb
[1996]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/atmega8_i2c-slave.sch
[1997]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/atmega8_i2c-slave.pcb
[1998]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/rpi-config.txt
[1999]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/rpi-rc.local
[2000]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/koncepcja_odczytu_di.c
[2001]http://www.opcode.eu.org//usage_and_config/smart_home/centralka_alarmowa/centralka_alarmowa.tgz
[2002]http://www.opcode.eu.org//usage_and_config/smart_home/manipulator_alarmowy/manipulator.sch
[2003]http://www.opcode.eu.org//usage_and_config/smart_home/manipulator_alarmowy/manipulator.pcb
[2004]http://www.opcode.eu.org//usage_and_config/smart_home/manipulator_alarmowy/manipulator-klawiatura.sch
[2005]http://www.opcode.eu.org//usage_and_config/smart_home/manipulator_alarmowy/manipulator_alarmowy.tgz
[2006]http://pl.wikipedia.org/wiki/RS-485
[2007]http://pl.wikipedia.org/wiki/Modulacja szerokości impulsów
[2008]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/sterownik_oswietlenia-4PWM.sch
[2009]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/sterownik_oswietlenia.tgz
[2010]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-PWM.sch
[2011]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-PWM.pcb
[2012]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-PWM.sym
[2013]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-PWM-conn.sch
[2014]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-no_PWM.sch
[2015]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-no_PWM.pcb
[2016]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-no_PWM.sym
[2017]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak-no_PWM-conn.sch
[2018]http://pl.wikipedia.org/wiki/RS-485
[2019]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/sterownik_oswietlenia-4PWM.sch
[2020]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/sterownik_oswietlenia-4PWM.pcb
[2021]http://www.elektroda.pl/rtvforum/viewtopic.php?t=44066
[2022]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak–PWM.sch
[2023]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak–no_PWM.sch
[2024]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_oswietlenia/modul-triak–PWM.sch
[2025]http://www.elektroda.pl/rtvforum/topic639040.html
[2026]http://www.elektroda.pl/rtvforum/viewtopic.php?t=53822
[2027]http://www.elektroda.pl/rtvforum/topic920408.html
[2028]http://www.elenota.pl/datasheet-pdf/31436/ON-Semiconductor/AN1048
[2029]http://www.elenota.pl/datasheet-pdf/6953/Fairchild/AN-3008
[2030]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/sterownik_LED_RGB.tgz
[2031]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-pcf8574.sch
[2032]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-pcf8574.pcb
[2033]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-mcp23008.sch
[2034]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-uln2803a-mcp23008.pcb
[2035]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-mosfet-uln2803a-mcp23008.sch
[2036]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-mosfet-uln2803a-mcp23008.pcb
[2037]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-8npn-pcf8574.sch
[2038]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-8npn-pcf8574.pcb
[2039]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/modul-i2c-8npn.sym
[2040]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/multiplexer_i2c.sch
[2041]http://www.opcode.eu.org//usage_and_config/smart_home/sterownik_LED_RGB/multiplexer_i2c.pcb
[2042]http://www.opcode.eu.org//usage_and_config/smart_home/audio_switch/switch_audio.sch
[2043]http://www.opcode.eu.org//usage_and_config/smart_home/niestabilizowany_zasilacz_buforowy/./niestabilizowany_zasilacz_buforowy.sch
[2044]http://www.opcode.eu.org//usage_and_config/smart_home/niestabilizowany_zasilacz_buforowy/elementy_podzasilaczy.sch
[2045]http://www.opcode.eu.org//usage_and_config/smart_home/niestabilizowany_zasilacz_buforowy/./podlaczenie_zasilania.sch
[2046]http://pl.wikipedia.org/wiki/Modulacja szerokości impulsów
[2047]http://www.opcode.eu.org//usage_and_config/smart_home/smart_home_story/sterownik_oswietlenia_din.pcb
[2048]http://www.opcode.eu.org//inne/photo/systemowy_sterownik_oswietlenia.0.
[2049]http://pl.wikipedia.org/wiki/EIA-485
[2050]http://pl.wikipedia.org/wiki/Universal Asynchronous Receiver and Transmitter
[2051]http://www.opcode.eu.org//usage_and_config/smart_home/smart_home_story/hub_rs_uC.sch
[2052]http://www.opcode.eu.org//usage_and_config/smart_home/smart_home_story/hub_rs_no_uC.sch
[2053]http://pl.wikipedia.org/wiki/Modbus
[2054]http://www.geda.seul.org/
[2055]http://en.wikipedia.org/wiki/GEDA
[2056]http://pcb.sourceforge.net/
[2057]http://geda.seul.org/tools/xgsch2pcb/index.html
[2058]http://www.lis.inpg.fr/realise_au_lis/kicad/
[2059]http://www.liquidpcb.org/
[2060]http://opencircuitdesign.com/xcircuit/
[2061]http://ktechlab.org/
[2062]https://github.com/marc-lorber/oregano
[2063]http://www.gnu.org/software/gnucap/
[2064]http://ngspice.sourceforge.net/
[2065]http://www.a-rostin.de/klogic/
[2066]http://ksimus.berlios.de/
[2067]http://qucs.sourceforge.net/
[2068]http://www.tkgate.org/

[2069]http://www.nongnu.org/uisp/
[2070]http://savannah.nongnu.org/projects/avrdude/
[2071]http://avarice.sourceforge.net/
[2072]http://avrprog.sourceforge.net
[2073]http://tihlde.org/ jonah/el/avrp.html
[2074]http://alioth.debian.org/projects/minicom
[2075]http://efault.net/npat/hacks/picocom/
[2076]http://jpnevulator.snarl.nl/
[2077]http://sourceforge.net/projects/gtkterm/
[2078]http://cutecom.sourceforge.net/
[2079]http://www.qcad.org/
[2080]http://en.wikipedia.org/wiki/LibreCAD
[2081]http://www.qcad.org/
[2082]http://pl.wikipedia.org/wiki/QCad
[2083]http://freecadweb.org/
[2084]http://www.pythoncad.org/
[2085]http://www.openscad.org/
[2086]http://www.gnome.org/projects/dia/
[2087]http://archimedes.iv.org.br/portal/downloads
[2088]http://www.graphviz.org/
[2089]http://gerbv.sourceforge.net/
[2090]http://ghdl.free.fr/
[2091]http://gtkwave.sourceforge.net/
[2092]http://www.icarus.com/eda/verilog/
[2093]ftp://ftp.cs.man.ac.uk/pub/amulet/balsa/other-software/espresso-ab-1.0.tar.gz
[2094]http://www.baudline.com/
[2095]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/demo.cir
[2096]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/demo.circuit
[2097]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/pcb.inc
[2098]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/gschem-symbole.tgz
[2099]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/gschem-symbole.notar
[2100]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/pcb-elements.tgz
[2101]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/pcb-elements.notar
[2102]http://www.gedasymbols.org/
[2103]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/pcb-show.tgz
[2104]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/pcb-show.notar
[2105] http://www.geda.seul.org/
[2106]http://pcb.sourceforge.net/
[2107]http://www.linuxfocus.org/English/December2004/article355.shtml
[2108]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/qcad-elektryczne.tgz
[2109]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/qcad-elektryczne.notar
[2110]http://en.wikipedia.org/wiki/Flowchart
[2111]http://en.wikipedia.org/wiki/State diagram
[2112]http://pl.wikipedia.org/wiki/Unified Modeling Language
[2113]http://dia-installer.de/shapes.html
[2114]http://www.hawlisch.de/devel/dia/index.html
[2115]http://code.google.com/p/jubarte/downloads/list
[2116]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/soft4electronic/Assorted.sheet
[2117]http://www.fuw.edu.pl/ pablo/pracownia/gpcard.ps
[2118]http://www.fuw.edu.pl/ pablo/pracownia/gnuplot.pdf
[2119]http://gnuplot.sourceforge.net/docs/gnuplot.pdf
[2120]http://www.gnuplot.info/docs/tutorial.pdf
[2121]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/atmega8_uart_spi.sch
[2122]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/atmega8_uart_spi.pcb
[2123]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/atmega8_uart_spi.sym
[2124]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/atmega8_dev_board.pcb
[2125]http://www.freemodbus.org/
[2126]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/Makefile
[2127]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/modbus.tar
[2128]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/modbus.c
[2129]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/libs4avr/port.h
[2130]http://www.opcode.eu.org//inne/photo/soft_start.0.
[2131]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/generator/generator.tgz
[2132]http://pl.wikipedia.org/wiki/Universal Asynchronous Receiver and Transmitter
[2133]http://pl.wikipedia.org/wiki/RS-485
[2134]http://pl.wikipedia.org/wiki/RS-422
[2135]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_RS/RS.sch
[2136]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_RS/RS.pcb
[2137]http://pl.wikipedia.org/wiki/RS-232
[2138]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_RS/RS232-via-RJ45.svg
[2139]http://yost.com/computers/RJ45-serial/
[2140]http://serwis.avt.pl/manuals/AVT530.pdf
[2141]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_poziomow_ttl/transoptory.sch
[2142]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_poziomow_ttl/transoptory.pcb
[2143]http://www.elenota.pl/datasheet-pdf/126007/Philips/74AHCT125

[2144]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_poziomow_ttl/konwerter_poziomow.sch
[2145]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_poziomow_ttl/konwerter_poziomow.pcb
[2146]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_USB-RS/usb-rs.sch
[2147]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_USB-RS/usb-rs.pcb
[2148]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter_USB-RS/usb-rs.gsch2pcb
[2149]http://pl.wikipedia.org/wiki/1-Wire
[2150]http://pl.wikipedia.org/wiki/I²C
[2151]http://pl.wikipedia.org/wiki/Serial Peripheral Interface
[2152]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter-interfejsy_elektroniczne/rs-i2c_spi_1wire_8bit.sch
[2153]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter-interfejsy_elektroniczne/rs-i2c_spi_1wire_8bit.pcb
[2154]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter-interfejsy_elektroniczne/eth-rs_i2c_spi_1wire_8bit.sch
[2155]http://www.opcode.eu.org//usage_and_config/toolbox4electronic/konwerter-interfejsy_elektroniczne/eth-rs_i2c_spi_1wire_8bit.pcb
[2156]http://www.elektroda.pl/rtvforum/viewtopic.php?t=964798
[2157]http://www.lochraster.org/etherrape/
[2158]http://www.elektroda.pl/rtvforum/viewtopic.php?t=1388652
[2159]http://www.opcode.eu.org//more_advanced/programing/c_cpp/ftdi_ds18b20.tar.gz
[2160]http://www.mediawiki.org/wiki/Extension:StringLoop
[2161]http://www.mediawiki.org/wiki/Extension:CategoryIndex
[2162]http://pushvideoclub.com/viewtopic.php?f=99&t=163&p=228
[2163]http://www.grain-media.com/html/documentation/GM8181_GM-2009-09.pdf
[2164]http://www.opcode.eu.org//usage_and_config/other_projects/Avtech_KPD674_DVR_firmware/KPD674AZ.jpg
[2165]ftp://ftp.spselectronics.pl/cctv/AVC_KPD_MDR/firmware/kpd-674z_tylko_do_plyt_z.zip
[2166]http://download.dvrtw.com.tw/GPL/076D_Series/arm-linux-2.6.tar.gz
[2167]http://code.google.com/p/gm8161-sdk-v161
[2168]http://mak.rsmu.ru/cam-api/avtech/CGI_v2.1.doc
[2169]http://dir.icm.edu.pl/pl/
[2170]http://www.icm.edu.pl/
[2171]http://www.uw.edu.pl/
[2172]http://www.opcode.eu.org//usage_and_config/other_projects/icm-dirop/icm-dirop-web.tgz
[2173]http://www.opcode.eu.org//usage_and_config/other_projects/icm-dirop/icm-dirop-web.notar
[2174]http://www.opcode.eu.org//usage_and_config/other_projects/icm-dirop/icm-dirop-prepare-book.tgz
[2175]http://www.opcode.eu.org//usage_and_config/other_projects/icm-dirop/icm-dirop-prepare-book.notar
[2176]http://www.dlib.vt.edu/projects/OAI/software/xmlfile/xmlfile.html
[2177]http://www.opcode.eu.org//usage_and_config/other_projects/icm-dirop/OAI-XMLFile-rrp.patch
[2178]http://www.opcode.eu.org//usage_and_config/other_projects/shell_phpmyadmin/index.php
[2179]http://www.opcode.eu.org//usage_and_config/other_projects/NutOS_IM_notify/nutos_jabber.tgz
[2180]http://www.linuxfocus.org/Polish/September2003/article304.shtml
[2181]http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-sterownik.sch
[2182]http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-sterownik.pcb
[2183]http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-matryca_12V.sch
[2184]http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop-matryca_5V.sch
[2185]http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/matryca_flux.pcb
[2186]http://www.opcode.eu.org//usage_and_config/other_projects/stroboskop_LED/stroboskop_LED.tgz
[2187]http://www.opcode.eu.org//usage_and_config/other_projects/sterownik_ogrzewania/mikrokontroler.sch
[2188]http://www.opcode.eu.org//usage_and_config/other_projects/sterownik_ogrzewania/mikrokontroler.pcb
[2189]http://www.opcode.eu.org//usage_and_config/other_projects/sterownik_ogrzewania/plytka_czolowa.sch
[2190]http://www.opcode.eu.org//usage_and_config/other_projects/sterownik_ogrzewania/sterownik_ogrzewania.tgz
[2191]http://mterm.w.interia.pl/
[2192]http://skalar.mcten.pl/
[2193]http://www.elektroda.pl/rtvforum/viewtopic.php?t=589928
[2194]http://www.elenota.pl/search.php?man=–wszyscy–\&ile=50\&start=0\&szuk=hd44780
[2195]http://pl.wikipedia.org/wiki/Public domain
[2196]http://pl.wikipedia.org/wiki/Licencja BSD
[2197]http://pl.wikipedia.org/wiki/Licencja X11
[2198]http://pl.wikipedia.org/wiki/Licencja Artystyczna
[2199]http://pl.wikipedia.org/wiki/Copyleft
[2200]http://pl.wikipedia.org/wiki/GNU General Public License
[2201]http://pl.wikipedia.org/wiki/GNU Free Documentation License
[2202]http://pl.wikipedia.org/wiki/Licencja Creative Commons
[2203]http://pl.wikipedia.org/wiki/Otwarte Oprogramowanie
[2204]http://pl.wikipedia.org/wiki/Wolne Oprogramowanie
[2205]http://en.wikipedia.org/wiki/Comparison of free software licences
[2206]http://pl.wikipedia.org/wiki/Request for Comments
[2207]http://www.w3c.org/TR/#Recommendations
[2208]http://xmpp.org/extensions/
[2209]http://pl.wikipedia.org/wiki/ASCII
[2210]http://pl.wikipedia.org/wiki/Unicode
[2211]http://pl.wikipedia.org/wiki/ISO 8601
[2212]http://pl.wikipedia.org/wiki/IEEE 802.3
[2213]http://pl.wikipedia.org/wiki/IEEE 1394
[2214]http://pl.wikipedia.org/wiki/IEEE 1284
[2215]http://pl.wikipedia.org/wiki/JTAG
[2216]http://pl.wikipedia.org/wiki/POSIX
[2217]http://pl.wikipedia.org/wiki/Universal Serial Bus
[2218]http://pl.wikipedia.org/wiki/The Open Group

[2219]http://pl.wikipedia.org/wiki/Single UNIX Specification
[2220]http://www.opengroup.org/onlinepubs/009695399/
[2221]http://www.rfc-editor.org/rfcxx00.html
[2222]http://www.rfc-editor.org/rfcsearch.html
[2223]http://www.isoc.org.pl/wiki/index.php/RFC
[2224]http://en.wikipedia.org/wiki/Out-of-band management
[2225]http://pl.wikipedia.org/wiki/Telewizja_cyfrowa
[2226]http://pl.wikipedia.org/wiki/DVB
[2227]http://pl.wikipedia.org/wiki/HDTV
[2228]http://pl.wikipedia.org/wiki/Multimedia Home Platform
[2229]http://pl.wikipedia.org/wiki/Set-top_box
[2230]http://pl.wikipedia.org/wiki/Ethernet
[2231]http://pl.wikipedia.org/wiki/H.264/MPEG-4_AVC
[2232]http://pl.wikipedia.org/wiki/PAL
[2233]http://pl.wikipedia.org/wiki/NTSC
[2234]http://en.wikipedia.org/wiki/ISDB
[2235]http://pl.wikipedia.org/wiki/ATSC
[2236]http://pl.wikipedia.org/wiki/Sieć lokalna
[2237]http://pl.wikipedia.org/wiki/Multicast
[2238]http://pl.wikipedia.org/wiki/Unicast
[2239]http://pl.wikipedia.org/wiki/Wideo na życzenie
[2240]http://pl.wikipedia.org/wiki/Telewizja satelitarna
[2241]http://pl.wikipedia.org/wiki/Telewizja kablowa
[2242]http://en.wikipedia.org/wiki/Test card
[2243]http://dvbzone.dvhk.pl/
[2244]http://www.dipol.com.pl/cyfrdip.htm
[2245]http://www.dipol.com.pl/bib67.htm
[2246]http://www.dipol.com.pl/schemat08.htm
[2247]http://www.dipol.com.pl/bib06.htm
[2248]http://pl.wikipedia.org/wiki/IPTV
[2249]http://adrianer.jogger.pl/2006/05/26/kajdanki-na-ogladanie-telewizji-koszmar-czy-nasza-przyszlosc/
[2250]http://en.wikipedia.org/wiki/Rec._601
[2251]http://en.wikipedia.org/wiki/Rec._709
[2252]http://pl.wikipedia.org/wiki/Komputerologia
[2253]http://pl.wikipedia.org/wiki/Błąd (informatyka)
[2254]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=470417
[2255]https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=425269
[2256]http://forum.psi-im.org/thread/4959
[2257]https://trac.videolan.org/vlc/ticket/9708
[2258]https://bugzilla.mozilla.org/show_bug.cgi?id=14328
[2259]http://dns.vanrein.org/srv/
[2260]https://github.com/aperezdc/ngx-fancyindex/issues/2
[2261]https://bugzilla.libav.org/show_bug.cgi?id=284
[2262]https://bugzilla.libav.org/show_bug.cgi?id=578
[2263]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=692876
[2264]https://trac.videolan.org/vlc/ticket/9707
[2265]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=440681
[2266]http://wiki.geda-project.org/geda:format_translation
[2267]http://savannah.gnu.org/bugs/?23030
[2268]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=257676
[2269]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=634005
[2270]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=484011
[2271]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=319244
[2272]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=603914
[2273]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=634007
[2274]https://bugzilla.icculus.org/show_bug.cgi?id=5045
[2275]https://bugzilla.icculus.org/show_bug.cgi?id=5054
[2276]https://bugzilla.icculus.org/show_bug.cgi?id=4523
[2277]https://bugzilla.icculus.org/show_bug.cgi?id=3769
[2278]https://bugzilla.xfce.org/show_bug.cgi?id=7714
[2279]https://bugzilla.xfce.org/show_bug.cgi?id=4887
[2280]https://bugzilla.xfce.org/show_bug.cgi?id=5605
[2281]https://bugzilla.xfce.org/show_bug.cgi?id=2266
[2282]https://bugzilla.gnome.org/show_bug.cgi?id=579964
[2283]https://bugs.kde.org/show_bug.cgi?id=206746
[2284]https://bugs.kde.org/show_bug.cgi?id=258169
[2285]https://bugs.kde.org/show_bug.cgi?id=169691
[2286]https://bugs.kde.org/show_bug.cgi?id=176069
[2287]https://bugs.kde.org/show_bug.cgi?id=34692
[2288]https://bugs.kde.org/show_bug.cgi?id=151672
[2289]http://pdfedit.petricek.net/bt/view.php?id=218
[2290]http://xorg.freedesktop.org/wiki/RadeonFeature
[2291]http://www.x.org/wiki/radeonhd
[2292]http://www.x.org/wiki/radeon
[2293]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=502030

[2294]http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=499018
[2295]https://retracile.net/wiki/VimBreakIndent
[2296]http://sqizit.bartletts.id.au/2011/01/31/vim-and-breakindent/
[2297]http://sourceforge.net/mailarchive/forum.php?thread_name=4DBA67D6.1060601%40gmx.net&forum_name=tmux-users
[2298]http://en.wikipedia.org/wiki/Copyright_Act_of_1790
[2299]http://pl.wikipedia.org/wiki/Domena publiczna
[2300]http://www.wolnakultura.org/ksiazka,25.html
[2301]http://koed.org.pl/dzien-domeny-publicznej/materialy/
[2302]http://www.krytykapolityczna.pl/Serwis-piracki/Czy-piraci-uratuja-kulture/menu-id-289.html
[2303]http://www.opcode.eu.org//archiwum/studia_elka/fg/Zagadnienie własności intelektualne na progu XXI wieku.pdf
[2304]http://theta.uoks.uj.edu.pl/ gszczepa/gszczepa/esr1iso2.htm
[2305]http://pl.wikipedia.org/
[2306]http://pl.wikipedia.org/wiki/Wikiportal:Informatyka
[2307]http://pl.wikipedia.org/wiki/Kategoria:Informatyka
[2308]http://pl.wikipedia.org/wiki/Kategoria:Telekomunikacja
[2309]http://pl.wikipedia.org/wiki/Kategoria:Elektronika
[2310]http://pl.wikipedia.org/wiki/Kategoria:Elektrotechnika
[2311]http://pl.wikibooks.org/
[2312]http://helionica.pl/index.php/Strona_główna
[2313]http://www.fuw.edu.pl/ rwys/kis/kis.pdf
[2314]http://www.fuw.edu.pl/ rwys/pk/pk.pdf
[2315]http://webster.cs.ucr.edu/AoA/
[2316]http://weirdscience.eu/
[2317]http://www.linuxfocus.org/
[2318]http://linux.skryptoteka.pl/
[2319]http://www.linux-magazine.pl/Magazine/Archive
[2320]http://dragon10.foxnet.pl/
[2321]http://tldp.org/
[2322]http://www.gentoo.org/doc/pl/articles/
[2323]http://pl.docs.pld-linux.org/
[2324]http://www.bsdguru.org/
[2325]http://www.ibm.com/developerworks/
[2326]http://sequoia.ict.pwr.wroc.pl/ witold/index.html
[2327]http://tsunami.kaniup.agh.edu.pl/dydaktyka.html
[2328]http://www.elfa.se/pl/fakta.pdf
[2329]http://www.sep.com.pl/opracowania.htm
[2330]http://www.sp1lop.internet.v.pl/ksiazka/
[2331]http://home.elka.pw.edu.pl/ pnazimek/index.php?lang=pl&page=39
[2332]http://wazniak.mimuw.edu.pl/
[2333]http://ebooks-it.org/
[2334]http://www.7thguard.net/
[2335]http://prawo.vagla.pl/node/7386
[2336]http://www.wolnakultura.org/
[2337]http://www.goodcopybadcopy.net/
[2338]http://www.pl.tomshardware.com/
[2339]http://pclab.pl/
[2340]http://www.debian.org/
[2341]http://dug.net.pl/
[2342]http://www.linuksowo.pl/
[2343]http://www.linux.com.pl/forum/
[2344]http://debian.linux.pl/
[2345]http://www.fsf.org/
[2346]http://www.gnu.org/directory
[2347]http://alioth.debian.org/
[2348]http://sourceforge.net/
[2349]http://www.bsd.org/
[2350]http://linux.pl/
[2351]http://www.unixy.pl/
[2352]http://www.linux.pc.pl/
[2353]http://www.linuxstorm.org/
[2354]http://openpoland.org/
[2355]http://www.tech.co.bydgoszcz.pl/
[2356]http://www.ppn.pl/
[2357]http://pl.wikipedia.org/wiki/b:Statyka budowli
[2358]http://pl.wikipedia.org/wiki/b:Wytrzymałość materiałów
[2359]http://www.meebo.com/
[2360]http://cb.vu/
[2361]http://www.traceroute.org/
[2362]http://gray-world.net/pl/pr_wsh.shtml
[2363]http://imex.sfh.pl/
[2364]http://zvoip.4.pl/
[2365]http://secviz.org/?q=image/tid/1
[2366]http://okvm.sourceforge.net/kvmoverip.html
[2367]http://commons.wikimedia.org/
[2368]http://pl.wikipedia.org/wiki/Wikipedia:Research_resources

[2369]http://en.wikipedia.org/wiki/Wikipedia:Public domain image resources
[2370]http://www.brickshelf.com/cgi-bin/gallery.cgi?f=49702
[2371]http://www.teleman.pl/
[2372]http://en.wikipedia.org/wiki/International Fixed Calendar
[2373]http://pl.wikipedia.org/wiki/Symmetry454
[2374]http://pl.wikipedia.org/wiki/World Calendar
[2375]http://pl.wikipedia.org/wiki/Licencja_BSD
[2376]http://pl.wikipedia.org/wiki/Licencja_X11